Multi-Robot Vision

Raphael Grech

March 2013

A Thesis submitted for the degree of Doctor of Philosophy

Faculty of Science, Engineering and Computing
Kingston University, London

EUROPEAN SOCIAL FUND
MALTA 2007-2013

The research work disclosed in this publication is partially funded by the Strategic Educational Pathways Scholarship Scheme (Malta). The scholarship is part-financed by the European Union - European Social Fund (ESF) under Operational Programme II - Cohesion Policy 2007-2013, "Empowering People for More Jobs and a Better Quality of Life".

Operational Programme II Cohesion Policy 2007-2013
Empowering People for More Jobs and a Better Quality of Life
Scholarships part-financed by the European Union
European Social Fund (ESF)
Co-financing rate: 85% EU Funds; 15% National Funds

Investing in your future

To my parents.
For their endless love, support and encouragement.

Abstract

It is expected nowadays that robots are able to work in real-life environments, possibly also sharing the same space with humans. These environments are generally considered as being cluttered and hard to train for. The work presented in this thesis focuses on developing an online and real-time biologically inspired model for teams of robots to collectively learn and memorise their visual environment in a very concise and compact manner, whilst sharing their experience to their peers (robots and possibly also humans). This work forms part of a larger project to develop a multi-robot platform capable of performing security patrol checks whilst also assisting people with physical and cognitive impairments to be used in public places such as museums and airports.

The main contribution of this thesis is the development of a model which makes robots capable of handling visual information, retain information that is relevant to whatever task is at hand and eliminate superfluous information, trying to mimic human performance. This leads towards the great milestone of having a fully autonomous team of robots capable of collectively surveying, learning and sharing salient visual information of the environment even without any prior information. Solutions to endow a distributed team of robots with object detection and environment understanding capabilities are also provided. The way in which humans process, interpret and store visual information are studied and their visual processes are emulated by a team of robots. In an ideal scenario, robots are deployed in a totally unknown environment and incrementally learn and adapt to operate within that environment.

Each robot is an expert of its area however, they possess enough knowledge about other areas to be able to guide users sufficiently till another more knowledgeable robot takes over. Although not limited, it is assumed that, once deployed, each robot operates in its own environment for most of its lifetime and the longer the robots remains in the area the more refined their memory will become. Robots should to be able to automatically recognize previously learnt features, such as faces and known objects, whilst also learning other new information. Salient information extracted from the incoming video streams can be used to select keyframes to be fed into a visual memory thus allowing the robot to learn new interesting areas within its environment. The cooperating robots are to successfully operate within their environment, automatically gather visual information and store it in a compact yet meaningful representation. The storage has to be dynamic, as visual information extracted by the robot team might change. Due to the initial lack of knowledge, small sets of visual memory classes need to evolve as the robots acquire visual information. Keeping memory size within limits whilst at the same time maximising the information content is one of the main factors to consider.

Acknowledgements

This thesis would not have been possible without the support of many people.

I would sincerely like to thank my supervisor, Professor Paolo Remagnino, for his guidance and support throughout this study, and for the confidence shown in me throughout the process.

I would also like to thank Professor Dorothy Monekosso and Professor Sergio Velastin, for their support, good advice and encouragement, especially in the initial stages of the Ph.D research project.

Thanks go to Professor Francisco Flórez-Revuelta for numerous hours of discussion and for sharing his insight in machine learning techniques.

Thanks also go to my fellow research students and research associates in the Digital Imaging Research Centre at Kingston University, for making the day bearable in the office and for reminding me that I was not alone in pursuing a journey into unexplored wilderness.

Thanks to all my family and numerous friends who endured this long process with me and lent me much needed moral support. Thank you for keeping my social life active in what was otherwise a rather solitary venture.

And last but not least, massive thanks go to my parents for their unfailing emotional and financial support. Thank you both for fostering my imagination, for nurturing my dreams, for giving me strength and courage to chase, and for the endurance to achieve.

List of Publications and Submissions

Journals

R. Grech, D. Monekosso, and P. Remagnino, "Building visual memories of video streams", *Electronics Letters*, vol. 48, no. 9, pp. 487 - 488, 26 2012.

M. V. Espina, R. Grech, J. S. Cope, F. Felisberti, S. Mannan, D. Monekosso, and P. Remagnino, "Robot visual memory maps: compact encoding of saliency", *Robotics and Autonomous Systems*, (Submitted).

R. Grech, F. Florez-Revuelta, D. Monekosso, and P. Remagnino, "Robot Visual Memories", *Image and Vision Computing*, (Submitted).

Book Chapters

M. V. Espina, R. Grech, D. de Jager, P. Remagnino, L. I. Iocchi, D. N. Monekosso, M. Nicolescu, and C. King, *Intelligent Paradigms in Security,* ser. INNOVATIONS IN DEFENCE SUPPORT SYSTEMS-3, 2011, ch. Multi-Robot Teams for Environmental Monitoring.

C. King, M. V. Espina, R. Grech, R. Mullen, P. Remagnino, L. I. Iocchi, L. Marchetti, N. D, D. N. Monekosso, and M. Nicolescu, *Handbook on Soft Computing for Video Surveillance.* Taylor & Francis, 2011, ch. Multi-Robot and Multi-Camera Patrolling.

Conferences

R. Grech, D. Monekosso, D. De Jager, and P. Remagnino, "A vision-based system for object identification and information retrieval in a smart home", in *Proceedings of the First international joint conference on Ambient intelligence,* ser. AmI'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 239 - 247.

R. Grech, F. Flórez-Revuelta, D. Monekosso, and P. Remagnino, "Robot teams: Sharing visual memories", in *International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2012.

Contents

List of Figures

List of Tables

List of Trademarks

Kinect® - Microsoft Corporation

Rovio™ - WowWee® Robotics Corporation

Xbox® - Microsoft Corporation

"If we knew what it was we were doing, it would not be called research, would it?"

Albert Einstein, *US (German-born) physicist (1879 - 1955)*

Chapter highlights:
A multi-robot platform capable to work in real-time and able to visually learn and share memories between robots in a compact and meaningful manner is developed. The main contributions are in the fields of:

- environment monitoring,
- handling of visual information, and
- information sharing.

1 Introduction

One of the biggest challenges in mobile robotics is full autonomy. In an ideal scenario, robots are deployed in a totally unknown environment and incrementally learn and adapt to operate within that environment. Over the last two decades considerable effort has been made in extending robot capabilities to be able to operate in human and other highly unstructured environments. These environments range from indoor, outdoor, underwater, air and even outer space. Moreover, the last decade has witnessed a realistic effort to embed intelligent systems in unmanned platforms. This thesis presents contributions towards the great milestone of having a fully autonomous team of robots capable of collectively surveying, learning and sharing salient visual information of the environment even without any prior information.

1.1 Aims and objectives

This research study aims to look at methods for the visual exploration and automatic interpretation of an uncharted environment by a team of robots. Cooperating robots are to successfully operate within their environment, automatically gather visual information and store it in a compact representation. The storage has to be dynamic, as visual information extracted by the robot team might change. The way in which humans process, interpret and store visual information will be studied and their visual processes will be emulated with a team of robots. The human process will be abstracted in an algorithm and employed on-board the robot platforms. The idea is to design a method using theories of visual understanding put forward by cognitive psychologists to embed a visual reinforcement process similar to the one used by humans to attend and recall relevant visual stimuli and ignore visual distracting elements of the scene. Such systems are of great value in applications where robotic platforms are necessary to solve a hard task without the help of humans. With the use of visual information, the knowledge about the environment will be enhanced by putting an understanding into what is within the environment. As an example, office blocks generally have the same geometrical layout, however, contents may vary between floors. Instead of just focusing on geometric map building, the idea is to also construct a graph of interesting scenes which are topologically linked. Assuming limited resources on the robot, an efficient way for the robot to learn the main scene differences and memorise them is needed.

As will be further discussed in Section 1.2.1, the work presented in this thesis forms part of a larger team project to develop a multi-robot platform capable of performing security patrol checks whilst also assisting people with physical and cognitive impairments to be used in public places such as museums and airports. The robots could be operated in three modes depending on the specific application. These mainly consist of assisted living, autonomous patrolling and human controlled operation.

- In the case of assisted living, the environment is somewhat structured by introducing identification tags, barcodes or RFIDs. Apart from being of use for robot operation, humans can also be equipped with hand-held assistive devices to help them read such labels and make use of the additional knowledge available in the environment.

- In the case of autonomous patrolling, such as security and surveillance applications, the robots operate autonomously. In such a scenario, especially where humans are present, the robots must be capable of self-navigation and reactive obstacle avoidance, due to the dynamic nature introduced by people moving around. Also, robots are able to automatically recognise and make sense of an entirely unstructured environment.

- In the third case, considering urban search and rescue (USAR) or area explorations as possible scenarios, humans such as security guards, can over-ride the robot's autonomous patrolling based on feedback coming from the robots. As robots are learning their new environment, they extract knowledge and relay it humans for further analysis. Humans could tele-operate the robot or instruct it on its next move or destination for better utilisation of the available resources. Moreover, in such a case, the environment may contain new features over time, so the robot learnt memory needs to evolve to reflect the new changes.

The work presented in this thesis aims to find a solution on how to amalgamate these areas onto a multi-robot platform, as suggested in Section 1.3, leading to the main contributions of environment monitoring, handling of visual information and information sharing, further described in Section 1.4. A detailed study on unsupervised learning techniques and feature extraction methods was carried out with the aim of finding a solution to producing a real-time and computationally feasible implementation for visual memories and saliency extraction. This research also relied on collaboration from Robot Vision Team[1] (RoViT) at Kingston University who conducted pilot tests using an eye tracker and provided data and preliminary results. Also, some state of the art methods especially in pattern detection, map building and navigation techniques were implemented and applied on the robots so as to provide a complete working multi-robot platform.

As will be seen in Section 2.3 topological maps have the advantage of compactly representing local variations, without keeping the raw visual information in memory. The basic idea is to design a method capable of handling complex visual information captured by one or more mobile cameras fitted on robot platforms forming part of a team. These camera equipped robots are used for surveying, monitoring and searching areas of interest whilst building a memory of the robot's environment. Both short and long term visual information, referred to as visual memories, are continuously elaborated to provide both a topology of landmarks and also building a knowledge base of objects or parts of the viewed scene and events deemed of interest. Visual information is encoded into a limited set of representative images on-line and with limited computational over-head. When there are several robots in an area, each doing its intended task, the visual information can be shared. One possible scenario could be that of having cleaning robots which are following a predefined route, and other robots tele-operated by security guards used for patrolling. If these robots are capable of creating visual memories, even though their intended applications are independent of each other, their memory can be shared, thus augmenting each others knowledge of other areas. A certain degree of redundancy is therefore expected which is considered as advantageous. These robots could reinforce the visual memory for salient points in the map whereas other low level features [5] will have memory decay. This would be somewhat similar to how ants reinforce their paths when foraging [6].

1.1.1 Challenges

Autonomy and **real-time operation** are two of the main challenges which have to be tackled to reach the intended aims. The team of robots has to navigate successfully and coordinate with its team members to understand and represent the scanned scene

[1]http://sec.kingston.ac.uk/research/research-groups/rovit/

as optimally as possible. Another challenge which has to be overcome is that of **fast deployment**. A method which could be deployed within a relatively short time (hours) would be desired.

1.2 Applications

The problem being studied could be related to several possible multi-robot applications and the proposed methods could be applied in a large number of domains. The operating environments could range from deep ocean sea to outer space. Some possible applications could include cleaning up a work site, performing search and rescue or extra-planetary exploration [7]. Also, the problem being tackled is not hardware specific and any kind of robot equipped with visual input can be used. Teams of robots are expected to gather information autonomously and collectively learn their potentially inhospitable environment in an autonomous and unsupervised fashion. Such as in the case of search and rescue, the acquired information could be used by humans or other robots alike when entering into the area and, depending on the situation, be warned in advance of any dangerous areas to avoid or important landmarks to be reached. Such robots could be scattered over large areas to explore alien environments which are totally unknown to the human race and of which there is no prior information. Unmanned areal vehicles(UAV) could be used to reach areas which would otherwise be inaccessible from the ground, e.g. by unmanned ground vehicles (UGV). One such example is that of robot forest monitoring and analysis, possibly in the case of controlling forest fires. The most widespread approach for forest analysis is that of using airborne laser-scanning data from a LiDAR and aerial photographs [8, 9, 10]. Another closely related application is that of automatically learning to identify and classify trees for deforestation and ecosystem study [11], wood production, three-dimensional (3D) treetop positioning, height estimation, species recognition, crown width estimation and stem diameter amongst others [12, 10, 13]. One method is by using colour and texture features [14, 8] together with machine learning techniques [14, 15] for the prediction of species-specific forest attributes.

Some more potential robotic application domains could include search and rescue missions and exploration missions in hazardous environments. At present, search dogs are the most useful tool to search for humans during search and rescue missions [16]. For such missions, a qualified dog needs at least 1.5 years for training and its effective working time is three years. Moreover, the dog cannot work for over two hours and continuous working time must not exceed 30 minutes [16]. At present search and rescue robots still struggle to reach these figures mainly due to the power storage on the robot, manoeuvrability and onboard sensors. This is however an active research area and it is expected to see some reasonable advances in the near future.

Another area of active research is in the field of ambient assisted living (AAL). At present, the use of guide dogs to help visually impaired people to navigate around is relatively common. For the same reasons mentioned above, and potentially additional benefits, robots are slowly but surely being introduced to help the elderly and less-able people. Some such robots include RIBA (Robot for Interactive Body Assistance)[2] and smart wheelchairs [17, 18].

Surveillance of public managed spaces is yet another area which could benefit from the study presented in this thesis. Such places where surveillance is of crucial importance and never enough would include airports, sporting venues and museums. In these environments, people need to feel safe without the feeling of being observed. Robots could be designed in such a way to allow people to interact with them. For humans to accept surveillance robots, they must be approachable, similar to police or security officers. There are several studies carried out on this, some of which will be discussed in Section 2.1.1.

1.2.1 The application

The work described in this thesis forms part of a larger project to develop a multi-robot platform capable of performing security patrol checks whilst also assisting people with physical and cognitive impairments to be used in public places such as museums and airports. When visiting such places, one would be aware that although the general layout of the environment does not change, the crowd density, flow and furniture layout change over time, with the expectation of clutter, complexity and unexpected events to happen. Therefore there is the need for robots to automatically adapt and learn their evolving environments with minimal human intervention.

Robots would generally require a previously built map of the area for efficient navigation within the area. If however, were the map not available (blueprint), the robots would need to be able to build a fairly accurate map of the environment using simultaneous localisation and mapping (SLAM) techniques and make it available to mobile security guards and other robots working in the same area. A method where robots are deployed and create a map of their surroundings in a relatively short time is suggested. Once this is done, the deployed robots can start performing their designated task, be it guidance, surveillance or whichever task they are assigned to do. In the meantime, using visual memories, robots can start learning the contents of the area they are operating in and storing it in a compact yet meaningful manner. Also, the acquired knowledge is shared between peer robots and also personnel who request information from the robots in order to guide the robots to specific areas.

Part of this project, supported by the U.S. Department of Homeland Security

[2]http://rtc.nagoya.riken.jp/RIBA/index-e.html

(U.S.DHS)[3] consists in emulating an airport security area and baggage collection. As it happens in border control, persons would enter the area, where a picture and other information will be gathered by the camera system operated by security personnel. These data can be automatically transferred throughout the security network and available to other security guards and robots for person identification. Security guards would be equipped with tablets and would be able to access information about people and robots in the area. A two-dimensional(2D) plan view of the security area is available and the tracked persons and robots are marked. The guard can tap on specific persons and their picture and other relevant information shows up. In an airport scenario, if a person was marked as requiring special assistance when they purchased the ticket, a robot goes to greet the person and guide them to the baggage collection point and then to the exit. Security guards also have the capability of controlling the robot remotely and send it to specific areas for further inspection just by tapping the desired position. The robot can capture images, acquire more detailed information, or even use speech to give instructions to persons within that area. In such a scenario the robot's tasks are to:

- guide users to their desired destination,
- interact with peer robots by sharing visual memories thus highlighting changes within their environment,
- share memories (obtained from other robots) and provide the user with what to expect at their destination, where another robot might be already there to take a handover,
- perform routine security checks during remaining/idle time, possibly controlled by security guards, who also have access to visual information and robot memory images.

For the robots to be able to perform as desired for the intended application, and reach the aims and objectives in a fully autonomous manner, the robots must be capable of performing the following tasks:

- To know their environment (Map)
- Interact with the user and environment (Obstacle avoidance / Human - Machine interaction)
- Detect (Face / Pattern detection) and Identify its user (RFID / Tag identification)

[3]U.S. Department of Homeland Security (U.S.DHS) Science and Technology Assistance Agreement No. 2011-ST-108-000021 awarded by the U.S. Department of Homeland Security.

- Learn and adapt to changes (Visual Memories)
- Identify important regions in the scene (Saliency)
- Coordinate with other robots for efficiency (Teamwork)

The items in brackets are the areas which need to be studied and developed in order to have a complete operational multi-robot platform. Figures 1.1 and 1.2 show snapshots from an actual implementation carried out at Kingston Hill Campus in August 2012. Figure 1.1 shows a robot performing face detection whist greeting a person requiring assistance (also using Text-to-speech) The sub-image in Figure 1.1 shows a snapshot from the robot camera with the face circled in blue. For the robot to be able to recognise and identify the specific person, additional information, possibly coming from an RFID tag would required. As one of several possible options, the robot could then guide the person to the baggage reclaim area as shown in Figure 1.2. The sub-image in Figure 1.2 shows the 2D map previously created using SLAM techniques by the robot. The robot autonomously navigates to its final destination whilst avoiding obstacles.

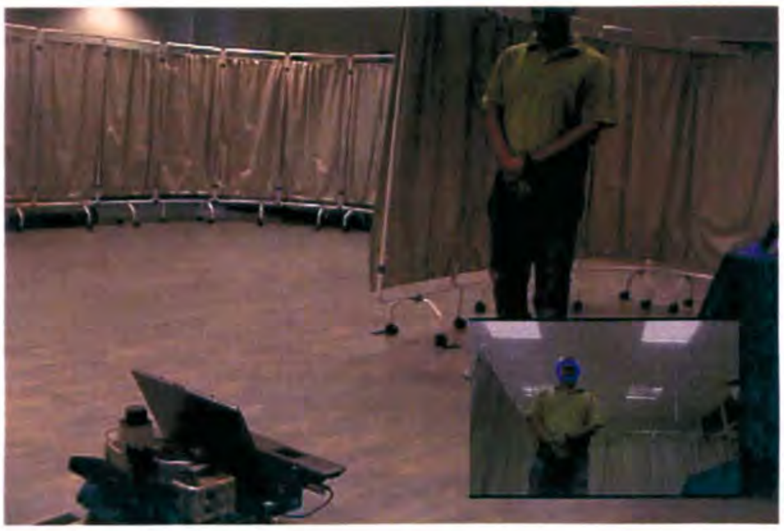


Figure 1.1: Face Detection

The multi-robot platform needed to handle such tasks will be described next. Further details on each item will be provided in more detail in the coming chapters.

1.3 Multi-robot platform

One of the initial requirements for the application, was that of designing a multi-robot platform to handle the tasks highlighted in Section 1.2.1. The following factors had to be considered:

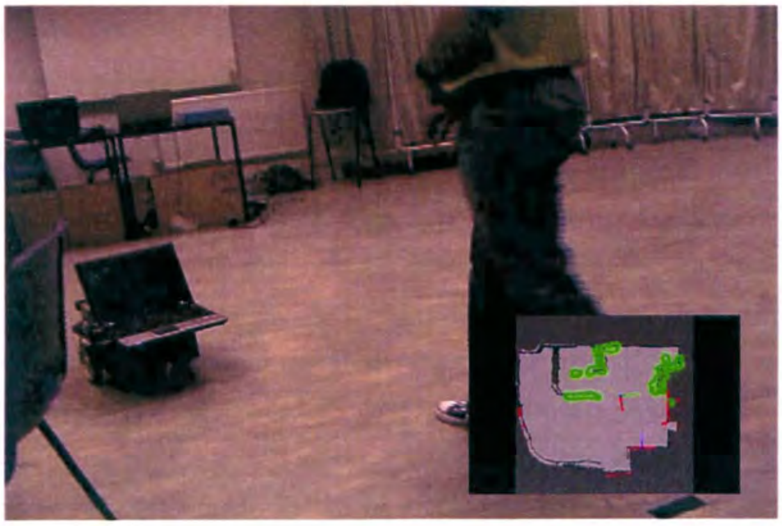


Figure 1.2: Person Guidance

- Platform design
  - hardware architecture
  - software framework
- Network Infrastructure
- Multi-Robot control
- Scene understanding (Computer Vision)
- Information fusion

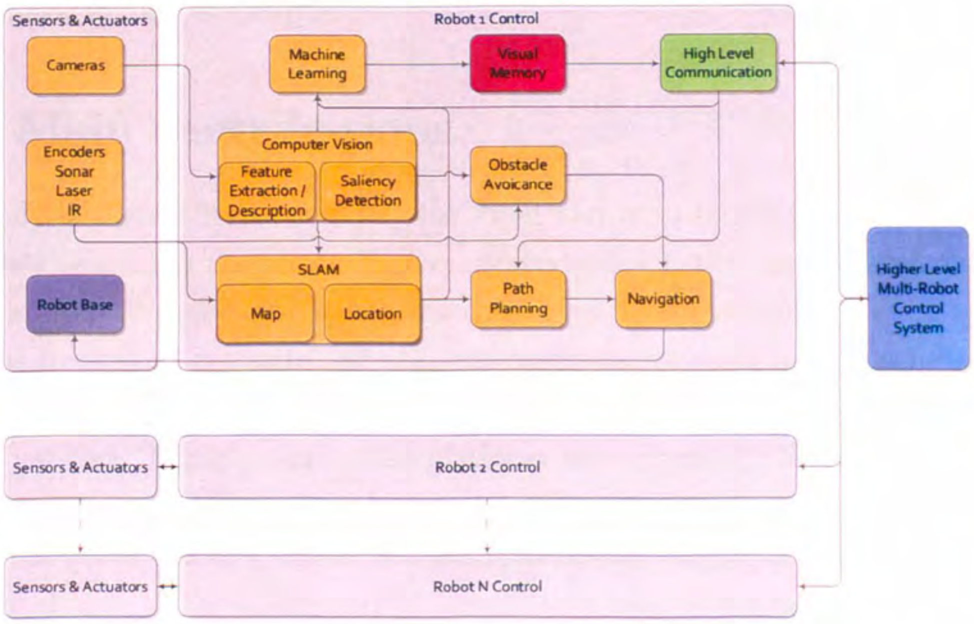


Figure 1.3: General Overview of a Multi-Robot Platform

Figure 1.3 shows the general overview of a multi-robot platform which would be able to work within an unstructured environment whilst learning its surroundings. The platform consists of a hardware architecture, operating system and software application frameworks. Each robot platform is divided in two main blocks which, in turn, are further divided into more specific ones. One block represents the sensors and actuators and the other block represents the robot's intelligent control. Information from the environment is gathered by sensors. Actuators are what make the robots able to react with the environment. Control is carried out on the robot's processor. Each robot would be able to scan the environment using its onboard sensors, such as a camera, identify interesting features and coordinate and share the information with the other team robots for combined learning. In the ideal scenario, robots should be able to be fully functional when operating independently and also able to handle information collectively when working within a multi-robot system (MRS). Items listed in the orange blocks are the necessary building blocks for an automated robot vision system capable of sensing, learning and reacting with the environment. Robots should be able to retain information that is relevant to whatever task is at hand and eliminate superfluous information. The red block reflects visual memories. Visual memory capability allows each robot to construct its own memory base of important features and visual information detected in the area. Operating in a multi-robot scenario, necessitates successful information handling and sharing between robots, represented by the green block. Moreover, when working in a centralised rather than distributed fashion, robots should be able to coordinate and accept commands coming from a higher level multi-robot control system (represented by a blue block) for collaboration, memory sharing and information fusion. Figure 1.3 will be further dissected into six sections and analysed further in Chapter 2.

1.4 Main contributions

As already discussed in Section 1.1, the main aim is to monitor an environment by collectively executing the most effective strategies for gathering the best quality information from it whilst overcoming the challenges highlighted in Section 1.1.1. Each robot has to explore, navigate and scan the environment using its own sensors, whilst sharing the visual and other relevant information with the other team robots over a wireless network. Using the proposed platform, as suggested in Section 1.3, this work focuses on advancing the state of the art in surveillance applications by enhancing multi-robot team learning based on computer vision techniques so as to provide a simple, real-time yet meaningful representation of the environment. Three main areas, namely environment monitoring, handling of visual information and information sharing, were tackled. These are described next.

1.4.1 Environment monitoring

Research in area monitoring involves various fields including exploration, identification, target tracking and localisation. Traditionally area monitoring relied solely on static sensory devices such as passive infra red (PIR) sensors and cameras (CCTV) and generally connected to a centralised control system. Acquiring visual information from moving cameras could be more beneficial. Apart from the standard fixed cameras, mobile robots can be equipped with cameras and use them to feed incoming streams to the central system but also use it to navigate or do other vision specific tasks. Moreover, mobile cameras can cover blind spots and share resources with the possibility for robots to acquire visual input from another robot's camera. As presented in [19] and [20], a practical solution for environment monitoring using robots equipped with cameras was provided. Using a group of mobile robots equipped with cameras has several significant advantages over a fixed surveillance camera system [20].

- The proposed solution can be used in environments that have previously not been equipped with a camera-based monitoring system. The robot team can be deployed quickly to obtain information about an unknown environment. Also robots would interact with each other to give the best coverage of the area and share the load [21]. The cameras installed on the robots, which can position themselves within the environment at the best vantage points, are used in order to best acquire the necessary information. This is in contrast with a static camera, which can only perform observations from a fixed view point.

- The robots in the team have the power to collaborate on the monitoring task and are able to pre-empt a potential threat.

- The robots could be equipped with additional, specialised sensors, which could be delivered at the appropriate place in the environment to detect, for example, the presence of high temperatures, such as in the case of a fire.

- The robot team can communicate with a human operator and receive commands about the goals and potential changes in the mission, allowing for a dynamic, adaptive solution.

Most video analytics is carried out on video streams acquired from stationary cameras. This generally allows for background subtraction and is also relatively simple, fast and effective. However, many cameras are not stationary eg. pan-tilt-zoom (PTZ) or even cameras fitted on cars, busses, handheld, head mounted, UAVs and robots. One of the biggest challenges with area monitoring using mobile devices is the fact that the area itself is unstructured and generally also shared with humans. Surveillance applications are generally required to operate in busy and crowded places such as commuter train stations, airports or shopping malls. Taking the next step of transferring the fixed cameras onto mobile platforms has a great impact on the complexity of the problem, especially considering the health and safety factors of having robots roaming around in the same environment with humans. The robots would therefore need to be endowed with map building, localisation, navigation and obstacle avoidance capabilities to operate within such a complex environment. These processes are essential for robots to be able to move freely, without bumping into team robots, people, surroundings or getting stuck in dangerous places.

1.4.2 Handling of visual information

Once the robots are able to operate successfully within their environment, the focus shifted on robot perception, with special emphasis on visual senses. A study on the identification of salient areas and pattern recognition within the environment was carried out together with an in-depth study on visual memory and a study on how visual information can be optimally represented in a compact manner for visual recognition. As presented in [22] and [23] the idea is to learn landmarks and build a topology for efficient representation of visual information. Processes are required to store visual information about the environment surrounding a robot in a compact representations and in re-usable format. A number of processes are involved with the extraction, fusion and manipulation of visual information. The extracted information is then used to solve either the recognition of a specific object, or the detailing of an area of space. Visual information is extracted using image and video descriptors using classifiers that can operate both online and in real-time. Available literature on human visual memory is studied with the aim to extrapolate the main processes involved with the extraction of image and video characteristics. Following the work suggested in [24] results obtained from controlled experiments using human subjects were used to indicate on how visual information can be extracted and interpreted. This is done by creating a bounded quantity of images gathered in separate clusters. This allowed the algorithm to create a flexible graph of nodes, each one holding a rich representation of a portion of space deemed of interest. It was then possible to recall and extract useful information from these images.

1.4.3 Information sharing

For better utilisation, the next step was that of getting the robots to successfully operate, communicate and collaborate within a heterogeneous environment. This not only allowed for the team of robots to manoeuvre within the same environment but also to collectively extract, share and elaborate visual information as a team. As presented in [25], the creation of a more holistic understanding of the environment was therefore possible. The data captured by the heterogeneous sensors are combined in a decentralised manner to improve the results of monitoring targets when there is a lack of scene coverage [24]. By integrating computer vision and robotics components using algorithms for the distributed visual accrual and fusion of visual information, it is shown that it is possible to monitor a large area using fewer sensors providing a scalable solution commensurate to the dimension of the monitored environment [20].

1.5 Thesis outline

The rest of the document is structured as follows: Chapter 2 provides an extensive review on the state of the art in multi-robot vision. Chapter 3 focuses on robot perception and how these techniques can be applied to make robots aware of their dynamic environment. Chapter 4 discusses visual memories and describes the method of implemented visual memories and how they can be utilised and extended over multiple robots leading to Chapter 5 where the methods suggested in earlier chapters are combined over a multi-robot scenario so as to have smart monitoring of complex public scenes. Conclusions are finally drawn in Chapter 6. Appendices A, B, C, D and E provide some additional material related to this study.

"Though human genius in its various inventions with various instruments may answer the same end, it will never find an invention more beautiful or more simple or direct than nature, because in her inventions nothing is lacking and nothing superfluous."

Leonardo Da Vinci, *Italian Renaissance polymath (1452 - 1519)*

Chapter highlights:
This chapter looks at the state of the art in multi-robot vision, focusing mainly on human-robot collaboration, robot mobility, machine vision and machine learning techniques.

2 Literature Review

Research in robotics is moving at a relatively fast pace. Starting from single fixed robots aimed at performing repetitive tasks within industry; these robots were successively transformed into mobile robots, not just confined into a specific location but also allowed to roam around for better resource utilisation. Further advancements allowed for groups of both homogeneous and heterogeneous robots to be created. This led to robots working within a team for a common aim. Some of these robots are nowadays also capable of interacting with humans [26]. One successful example where robots are allowed to roam, work in a team and interact with humans is the Kiva warehouse automation system[1],[2]. It is therefore expected that, in the foreseeable future, collaboration with humans will become common practice and an everyday experience for many. This necessitates communication between robots and humans carried out in a human understandable manner. Three important aspects which need to be addressed for a successful autonomous multi-robot and human interaction include:

- Operators must be able to quickly "grasp" the situation without being overwhelmed by unnecessary information.
- Robots in the system must act as an autonomous coordinated team: no direct human supervision should be required.
- Humans must be able to interact with robots in a natural way.

[1]http://www.kivasystems.com/solutions/
[2]http://www.youtube.com/watch?v=3UxZDJ1HiPE

This chapter provides a review on the state of the art in multi-robot vision. Figure 1.3 presented in Chapter 1 will be dissected into sections and analysed in further detail. Various areas in the robotics realm including robot mobility, computer vision and machine learning techniques are explored. Ways for team robots to interpret incoming data from sensors or from other robots through collaboration over communication networks are also studied.

2.1 Robotics

Robots can be generally described as electromechanical devices consisting of a physical body with sensors for feedback, mobility to be able to move within the environment, and reasoning capabilities to intelligently connect sensing with action [27]. Put in other words, an "intelligent" robot is one that can operate in a non-static environment and deal with uncertainties in sensors and effectors. Some intelligent robots can also improve their performance through learning and adaptation [27]. Today's robots can only perform highly specialised tasks, and their operation is constrained to a narrow set of environments and objects. Moreover, the majority of the world's eight million service robots are toys or drive in preprogrammed patterns to clean floors or mow lawns, while most of the 1 million industrial robots repetitively perform preprogrammed behaviours to weld cars, spray paint parts, and pack cartons [28]. It is a widely accepted fact that robots are a huge success story within this sector. Robots were originally developed for an industrial setting. These are confined within a structured and tailor made environment and are programmed to repeatedly do the same task to provide fast, efficient, repeatable, accurate and reliable assembly of consumer products. Robotics research is a field which involves various engineering challenges encompassing several disciplines such as electrical and electronic engineering, mechanical engineering and structural design, control engineering, software engineering and computer science. Considerable effort is being made in trying to overcome these challenges and extend robots to be able to operate in human and other highly unstructured environments. These environments range from indoor, outdoor, underwater, air and even outer space. Such environments pose far greater challenges when compared to their structured counterpart and that is what makes robotics an interesting subject for research. These environments also led to the use of new robot designs such as wheeled, legged, diving and even flying robots. Additional material on various robot types can be found in Appendix A, Section A.1.

Possibly, the most challenging of environments is that where both humans and robots share the same environment and possibly interact; with human safety being of highest concern. Part of the success in human-robot interaction (HRI), further discussed in Section 2.1.1, is related to the capability of the robots to sense and perceive their environment correctly using adequate sensors. As will be further seen in

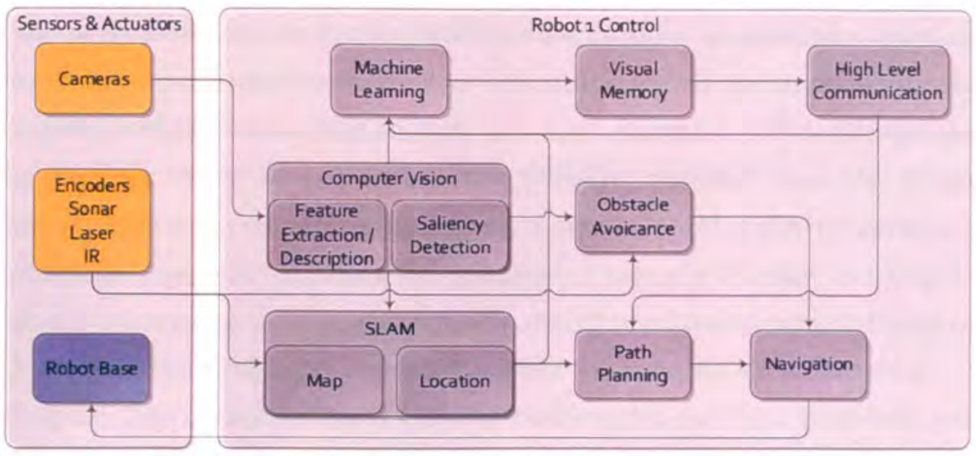


Figure 2.1: Sensors and actuators

Section 2.1.2, from the bionic aspect, the animals in nature sense the outside living space state and environment changes through various sensing apparatuses. According to the information gathered by sensing apparatuses, they dispose of the information comprehensively and reach the judgements. This process is referred to as multi-sensor information fusion [29]. Referring to Figure 2.1, robots within the natural world are expected to sense the environment and react to unexpected events accordingly. Several sensors have been created and used in the robotics environment to capture sound, vision and motion made by humans for the robot to be able to interact with its users in a more natural way. Some of these sensors are described in more detail in Appendix A, Section A.2

In some situations, a robot also needs to incrementally learn certain tasks. This further introduced new research concepts in the field of robotics such as environment sensing, obstacle detection, reactive control, real time planning and incremental learning techniques. For a mobile robot application to be successful it has to operate in real time with minimal human intervention. Efficient robot control, environment sensing, navigation, localization, exploration, map building, communication and sharing of information between robots are some of the key factors to consider when working with mobile robots. These will be explained in more detail in the coming sections.

2.1.1 Human - robot interaction (HRI)

Apart from humans and robots being increasingly used within the same workspace, these have to interact together in order to accomplish the overall mission goals [30]. Goodrich and Schultz [31] provide an in depth survey on HRI. They mention several problem domains for such area of study. Some of these include search and rescue, assistive and educational robotics, entertainment, military and police, space exploration, unmanned aerial vehicles (UAV) reconnaissance and unmanned underwater vehicles (UUV) applications [31]. Singer and Akin [30] carry out a survey of the field of collaborative human and robot team performance metric models, and examine existing overall team quantitative performance models to determine which are more applicable to future human and robotic space exploration missions. Murphy [32] describes how robots are being used for urban search and rescue (USAR) and discusses the human-robot interaction (HRI) issues encountered in such a scenario. Murphy presents a synopsis of the major HRI issues in reducing the number of humans it takes to control a robot, maintaining performance with geographically distributed teams with intermittent communications, and encouraging acceptance within the existing social structure. Nourjou et al. [33] state that coordination is the key challenging problem that field teams face in USAR emergency response because of the geographic and uncertain environment. They propose an efficient approach that allows humans to collaborate with coordinator assistant agents so as to assign tasks to teams both in time and space, efficiently and sufficiently in a way to optimize their coordination. Singer and Akin [30] analyze the performance of the heterogeneous teams to enable comparison between different team configurations.

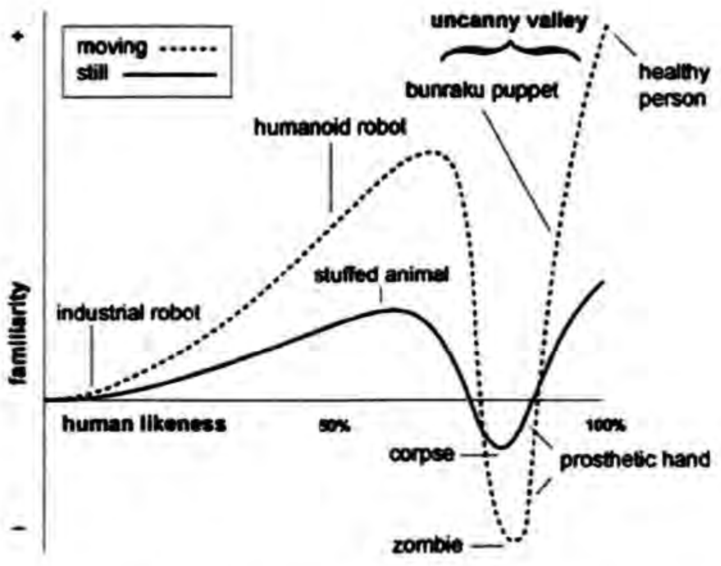


Figure 2.2: Mori's uncanny valley [1]

One of the most popular theories applied in the field of HRI is the uncanny valley theory [1], shown in Figure 2.2. This theory relates human-likeness of a robot with the level of familiarity evoked in the person interacting with the robot. Mori hypothesized that, as robots appear more and more human-like, people's familiarity with them increases until a point where this relationship ceases. Beyond this critical point, the appearance of the robot increases in human-likeness but the appearance no more evokes a feeling of familiarity. The robot instead is perceived as strange in appearance. Although the concept of the uncanny valley originated in robotics, it has generated higher levels of serious consideration in other areas related to human-like objects such as dolls, masks, facial caricatures, avatars in virtual reality and characters in computer graphics movies [34]. Beer at al. [35] carry out a study on robot acceptance in HRI with the goal to identify potential factors that predict user acceptance of personal robots. Amongst others they look into the robot's autonomy level, social capability and appearance. They present a table of specific tasks younger and older adults would want a robot to perform. Excluding the "other tasks" at (11%), the highest ranking tasks were "cleaning/chores" (35%), "security" (10%) and "physical aiding" (9%). The remaining lower ranking tasks amount to 44%. Given that Kingston University has strong links with visual surveillance groups and nursing / ambient assisted living (AAL), the research provided in [35] further suggests the applicability of this study in having a multi-robot system to assist humans in surveillance and assistance to people requiring special needs.

2.1.2 Biological inspiration

Nature (Life) evolved over millions of years, adapting and perfecting its living organisms. The way in which animals and plants successfully manage to survive and interact with each other is fascinating. It is therefore only wise to try to understand how such creatures operate and try to mimic them on artificial machines. Imperfections and mutations in nature lead to evolution. From a biological point of view nature evolved in a way to allow living organisms to interact with one another either by collaboration such as in ants and bees for foraging, flocking and schooling in birds and fish for protection, wolves and lions for hunting, and also symbiosis between different species for successful survival [36]. This inspired researchers, both in the academia and industry to look at these interactions within nature in order to replicate these on robots.

Especially in the case of multi-cellular organisms, each organism is composed of several organs which in turn are further composed of even smaller specialised cells. These specialised cells are generally unable to survive by themselves naturally but they successfully thrive when they coordinate together, evolving into something which is able to carry out complex tasks in a very efficient manner. One such organ is the brain. This is considered as being one of the most complex organs within the living body and consists of numerous, if not uncountable neuron cells. A similar approach consisting of artificial neurons was developed and applied for visual memories, as will be further described in Chapter 4.

The most commonly studied areas in distributed multiple robots generally include Swarms [37, 38], Cell Structures [39] and Genetic algorithms [40]. Social insects are worth copying because they have no centralised control but "system-level functioning is robust, flexible and scalable"[37]. By replicating simple local control rules used by various biological societies such as ants, bees and birds multi robot systems are able to flock, disperse, aggregate, forage, and follow trails [41]. Some of the advantages include:

- Robustness: "the swarm robotic system should be able to continue to operate, although at a lower performance, despite failures in the individuals, or disturbances in the environment ". The robustness of biological swarms can be attributed to:
  - Redundancy: "any loss or malfunction of an individual can be compensated by another one".
  - Decentralised control: destroying parts of the system will not stop the system as a whole from functioning.
  - Simplicity of the individuals: individuals are simpler, making them less prone to failures.
  - Multiplicity of sensing: "distributed sensing by large numbers of individuals can increase the total signal-to-noise ratio of the system".
- Flexibility: the system can generate solutions to different tasks. Ants for example, can perform several tasks such as chain forming, foraging and sorting. "Swarm robotic systems should also have the flexibility to offer solutions to the tasks at hand by utilizing different coordination strategies in response to the changes in the environment".
- Scalability: the system can operate under a wide range of group sizes.

As will be further discussed in Section 2.5, the above advantages however might come at the cost of lesser efficiency, increased cost and communication issues. Even though the individual entity cost and complexity may be less, determining how to manage the complete system may be more difficult and complex, because of the lack of centralized control or of a centralized repository of global information. Moreover, increasing the number of entities can lead to increased interference between entities, as they must act without complete knowledge of the other entities' intents [7].

Communication in nature can take various forms. The most common include the use of sounds, movement and chemical interactions. Some more advanced species elaborated sounds to create semantics leading to a language. Same for body movements by creating expressions commonly understood by the population. Some other species release chemicals as trails and signals thus leading to indirect coordination between peers. This is known as stigmergy. Stigmergy was first observed in social insects [42]. Ants, for example, exchange information by laying down pheromones on their way back to the nest when they have found food, which other ants will eventually follow. Subsequent similar actions by other ants will then reinforce the trail leading to the spontaneous emergence of coherent, apparently systematic activity. Bechon and Slotine [43] used a method known as Quorum sensing to robustly synchronize a group of humanoid robots, and to demonstrate the approach experimentally on a choreography of 8 robots. Quorum sensing is a system of stimulus and response correlated to population density. Many species of bacteria use it to coordinate gene expression according to the density of their local population. In similar fashion, some social insects use quorum sensing to determine where to nest.

As will be seen in Section 3.1, some organisms, including humans have perfected their capability of handling visual information. The key attentional mechanism, known as saliency detection, facilitates learning and survival by enabling organisms to focus their limited perceptual and cognitive resources on the most important areas within their environment. Parker [41] suggests that competition, such as that found in animals with higher intellect, including humans, can be applied for multi-robot systems. One such application is that of multi-robot soccer. Moreover, studying cooperation techniques within animals working in packs, such as wolves, has generated advances in cooperative control, more commonly known as predator - prey systems. Robot cooperation will be further discussed in Appendix Section A.6.3.

2.2 Robot vision

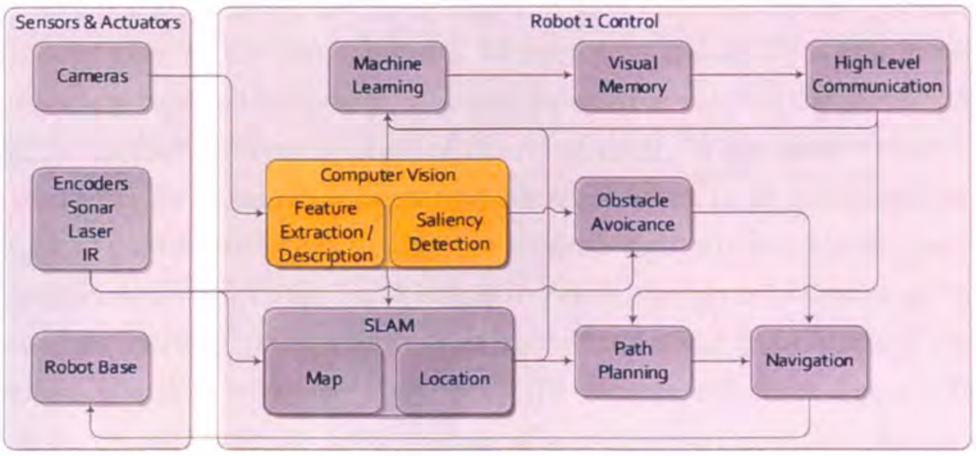


Figure 2.3: Computer vision

Robot vision and image processing play critical roles in the fast emerging robotics market, offering great opportunities for researchers, robot manufacturers, and vision component vendors [44]. As defined by the Robot Vision Group, Imperial College London[3], robot vision is related to the study of real-time computer vision techniques applicable to robotics or other demanding real-world, real-time applications. Moreover, Batchelor [45] states that machine vision and computer vision are not as synonymous

[3]http://www2.imperial.ac.uk/robotvision/website/php/

as one might expect even though they both refer to artificial vision. Computer vision is the science of analysing and processing digital images whereas machine vision is the application of computer vision to industrial tasks. Moreover, in machine vision it is generally accepted that the environment is more structured when compared to computer vision which tries to tackle problems in real world unstructured environments. Embedding computer vision technology in mobile robot platforms (see Figure 2.3), provides a more holistic view of a scene and augment the human visual experience from a new, so far little explored perspective. When one has to work with vision as the main sensorial input for the robot, various factors have to be considered. Sensing using vision still has lots of unsolved problems. Generally so far, its use is still restricted to a relatively structured and controlled environment. Ongoing research exists within the computer vision community to successfully identify the same object within the environment under varying illumination, perspective, motion, occlusion, orientation and distortion, to mention some. When using vision on robot systems, special attention has to be focused on ensuring that there will be no situation where the robot can get into a blind spot and unable to 'see' its environment due to physical / kinematic restrictions. This highly depends on the task required and the navigation plan intended for that particular robot. One has to consider the limitations of the cameras being used and when best to use one camera rather than another. The basic decision mainly revolves around monocular cameras when just one standard camera is used, stereo cameras when two fixed monocular cameras are used together and omnidirectional cameras, which provide a 360^o view of the environment. Moreover, it has to be borne in mind that robot motion can be characterised by the constraints that restrict the motion, generally caused by the mechanical construction of the robot itself. When measurements need to be extracted from an image, it is important for the camera to be compensated for lens distortions and calibrated by calculating the intrinsic and extrinsic camera parameters. Intrinsic parameters refer to the focal length in pixels, the principal point and the skew coefficient. The extrinsic parameters reflect the rotation and translation of the camera of a point on the grid reference plane onto the camera reference frame. The most common form of calibration is with the use of a calibration grid [46]. Recent studies also look into self calibrating techniques [47]. The two main stages in digital image processing for robots include [48]:

- preprocessing, which is a data preparation step for contrast enhancement, noise reduction or filtering
- feature extraction, for retrieving non-redundant and significant information from an image. This operation is targeted at achieving time efficiency at the cost of data reduction followed by object detection, localization and recognition, which determine the position, location and orientation of objects.

More material on image processing and comparison techniques is provided in Appendix Section A.3

2.2.1 Object detection

In literature, one finds several object detection techniques. Some techniques used include principal component analysis(PCA), neural networks(NN), support vector machines(SVM), Hough transform(HT), geometrical template matching(GTM) and colour analysis [2]. As will be seen in Section 2.4 various machine learning techniques exist. One of the most notable successes of application of machine learning for object detection in computer vision is the Viola-Jones face detector [49, 50]. Under the machine learning paradigm, this detector is said to be of the supervised learning type. For supervised learning, the system needs first to be thought what it is expected to recognise. This is done through an iterative process by teaching the algorithm what and what not the object of interest to detect is. This detector is slow in training, however is very fast in detection and can be used in real-time robot applications. Apart from faces, this technique works also fairly well on other, mostly rigid, objects that have distinguishing views [46]. The Viola-Jones detector was used on the robots first to detect faces and then was later adapted for tag detection as further described in Section 3.2.1.

Viola-Jones propose using a cascade of classifiers by training a set of weak Haar like classifiers. The weak classifiers are then combined to make a strong classifier. The classifier builds a form of Adaboost organised as a boosted rejection cascade where typically histogram- and size-equalised images patches are presented to the classifier, which are then labelled as containing (or not containing) the object of interest. The weak classifiers that it boosts in each node are decision trees that often are only one level deep. Each classifier is trained on a single feature so the relation between the classifiers and the features in one to one. The set of features used to train the weak classifiers are selected automatically using an Adaboost meta-learning algorithm. The learning algorithm to learn each classifier is independent on the selected feature. The comparison is very simple. If the value of a particular feature is above a threshold it would indicate the presence of an object, and not if otherwise [46]. The first stages of the cascade use only few features while the complexity increases in later stages. A data point weighting distribution is initialised, telling the algorithm how much misclassifying a data point will "cost". During boosting, as the algorithm progresses, the cost will evolve so that weak classifiers trained later will focus on the data points that the earlier trained weak classifiers tended to do poorly on. [46].

During the detection phase a scanning window is used to scan the input image in different locations and scales. Selected features are extracted from each window and fed to the final classifier which decides if an object is present in that window or not. True class detection is declared only if the computation makes it through the entire cascade. As in the case of faces in an image, the object of interest generally comprises of a small percentage of the image. For each node, a "not in class" result at any stage of the cascade terminates the computation, and the algorithm then declares that no face exists at that location. Rejection cascades can therefore greatly reduce total computation because most of the regions being searched for terminate quickly in a nonclass decision [46].

2.2.2 Depth and 3D measurement

Recently there has been a great interest in processing data acquired using depth measuring sensors due to the availability of cheap and efficient RGB-D cameras. Before the introduction of Kinect[4], most researchers used methods such as parallax and epipolar geometry in stereo cameras to obtain depth information. The most commonly used stereo cameras are those provided by Point Grey[5] and Videre[6]. Stereo correspondence was however generally considered as being both time and computationally intensive and only systems equipped with high processing capabilities could obtain usable results. Moreover, cameras mounted on moving robots require specialised engineering to approach the needed precision [51]. The Kinect camera developed by Prime Sense and Microsoft has considerably changed the situation, providing a three-dimensional(3D) camera capable of providing an RGB and depth image at a very affordable price. The 3D reconstruction problem is solved based on the understanding on multiple view geometry and generally consists of 3 steps [52]:

- Feature Detection and Tracking
- 3D reconstruction
- Normalization and upgrading to true structure.

Feature detection techniques are methods of identifying points of interest which could be compared between similar images. Visual descriptors outline the detected visual features in images or videos. The term visual descriptors sometimes also refers to algorithms or applications that produce such descriptions. They describe elementary characteristics such as shape, colour, texture, location or motion, among others. Ideally, feature descriptors should be scale, rotation, illumination and viewpoint invariant. To achieve an invariant description is particularly complex in the case of visual landmarks, since the appearance of a point in space varies greatly with viewpoint changes. In consequence, the data association problem becomes hard to solve. When there is a

[4]http://www.xbox.com/en-GB/KINECT
[5]http://www.ptgrey.com/products/stereo.asp
[6]http://www.videre.com (possibly discontinued)

region of no-texture, e.g. white wall or a grassy land seen from afar, trackers simply fail. On the other hand, when there is repeating structure, e.g. bricks on a brick wall, and when no global context is used for tracking, there is no method of finding true matches. Therefore, the requirements of the problem are 2-fold: there should be features and they should be distinct [52].

3D reconstruction can be performed to extract the robot location and scene structure. When two cameras (or a single moving camera) view a 3D scene from two distinct positions, there are a number of geometric relations between the 3D points and their projections onto the 2D images that lead to constraints between the image points. These relations are derived based on the assumption that the cameras can be approximated by the pinhole camera model and is known as epipolar geometry [53]. The pinhole camera model is the most widely used camera model mainly due to its simplicity but which none the less provides enough accuracy in calculations. The essential matrix contains information about the translation and rotation that relate the two cameras in physical space and the fundamental matrix contains the same information as the essential matrix in addition to information about the intrinsic parameters of both cameras. The fundamental matrix can be used to simplify the matching process between the viewpoints and to get the camera parameters in active systems where optical and geometrical characteristics might change dynamically depending on the image scene [54]. This subject has been comprehensively addressed in classic books by Hartley and Zisserman [53], Faugeras and Long [55] and Ma et al. [56]. 3D reconstruction algorithms are often designed to provide different tradeoffs between speed, accuracy, and practicality. In addition, even the output of various algorithms can be quite different. Lu et al. [57] survey a number of 3D reconstruction algorithms that exploit motion parallax. Fusiello [58] provide a review on 3D reconstruction techniques using a single camera with unconstrained motion and unknown parameters, highlighting auto calibration methods.

2.3 Robot mobility

One very important part in robot mobility is knowing where the robot stands within a map. This is known as localisation and is the ability to get one's position and orientation, correctly within the map. Mapping is the capability of the robot to create either a metric or topological map of its surroundings. Navigation is the capability to trace a route from point A to point B. This requires the determination of one's own position and to be able to plan a path towards some goal location [59]. Path planning can be therefore defined as devising the route for Navigation. This usually requires the use of a representation of the environment (a map), and the ability to interpret that representation. Topological maps are simplified maps where unnecessary details are removed and only vital information is maintained. These maps lack scale, distance and direction and are subject to change and variation, but the relationship between points is stored much as the tube map retains useful information despite bearing little resemblance to the actual layout of the underground system. This contrasts with metric maps where distance and direction are plotted accurately and most, if not all, of the acquired data is maintained. In the case of an application where no objects are specified, then change will be the main drive of the algorithm.

When working in an unknown or unstructured environment the map is not available and the robot must be capable to explore unknown and unvisited areas. In order to chart a map, map building and localisation have to be carried out simultaneously whist exploring the environment, leading to what is known within the robotics community as Simultaneous Localisation and Mapping (SLAM).

Although complementary, exploration is a totally different issue from SLAM. SLAM algorithms do not consider the computation of the movements that need to be performed by the robots since this is generally considered as a different problem denoted as exploration [60]. Generally, when a robot or a team of robots explore an unknown environment and build a map, the objective is to acquire as much new information as possible with every sensing cycle, so that the time needed to completely explore it is minimized [42]. Exploration algorithms try to optimize the paths to be followed such that the area to be explored is covered in the fastest possible time and avoid revisiting already known areas. This however causes a problem to SLAM as SLAM needs these places to be re-visited to increase the level of confidence. Extensive work has been carried out on robot mapping of the environment [61, 62, 63, 64]. Thrun [61] provides a comprehensive introduction into the field of robotic indoor mapping. Durrant and Whyte [62, 63] also provide a good explanation on the topic, describing and comparing various probabilistic techniques, as they are applied to mobile robot mapping problems. SLAM is discussed in more detail in Section 2.3.2.

2.3.1 Navigation, path planning and obstacle avoidance

As already highlighted earlier, and referring to Figure 2.4, navigation requires the determination of one's own position and to be able to plan a path towards some goal location using path planning techniques based on a map representation of the environment. While humans are able to navigate quite well based only on visual information, images usually require huge computer processing power. This means that for real-time robot operation, visual information is often avoided. Other sensors, such as sonar or laser range finders, provide accurate information at a much lower computational cost [65]. Albeit, visual navigation and localisation are active area of research. One technique could be that of using machine learning techniques for robot visual memories

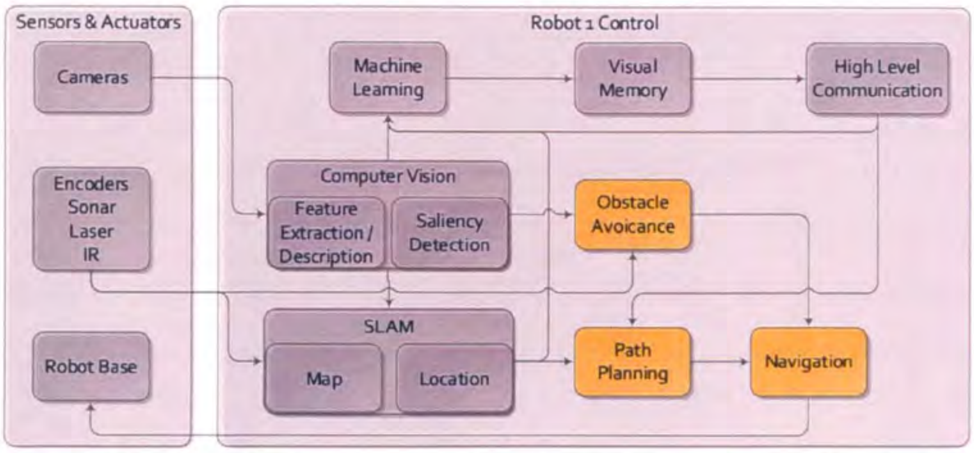


Figure 2.4: Path planning, obstacle avoidance and navigation

which is further discussed in Section 2.4. Navigation based on visual memories is very common among humans [66]. Moreover, when planning long trips, more sophisticated representations of the environment are created. Such representations could include topological maps where connections between paths are easily noted. The engineering and computing communities use such research as inspiration to carry on applied research to emulate visual memory onto machines.

In an unstructured environment the planned path will only be based on the partially known map. Path planning gets trickier when the physical robot constraints, as those mentioned in Appendix Section A.1, are taken into account. In the case of an unknown area exploration is necessary. With an unknown map the robot has to explore its surroundings and be able to navigate in real time by devising its best way in a reactive, rather than pre-planned, manner. This therefore requires the introduction of obstacle avoidance capabilities. Obstacle avoidance is the ability to detect the presence of obstacles between the robot's initial position and some goal position to be reached, and devising a suitable way to circumvent the obstacle. DeSouza et al. [67] provided a survey on vision for mobile robot navigation.

2.3.2 Simultaneous Localisation and Mapping (SLAM)

As already mentioned, for the robot to be able to navigate successfully within the environment a map needs to be constructed. Robots would use SLAM techniques, depicted in Figure 2.5, to build a map whilst locating themselves within that map. This would generally be a geometrical map representing the free and occupied areas in which the robot can eventually navigate or not. In SLAM, both the trajectory of the platform and the location of all landmarks are estimated online without the need of any a priori knowledge of location. Therefore the SLAM problem can be separated into two parts [60]:

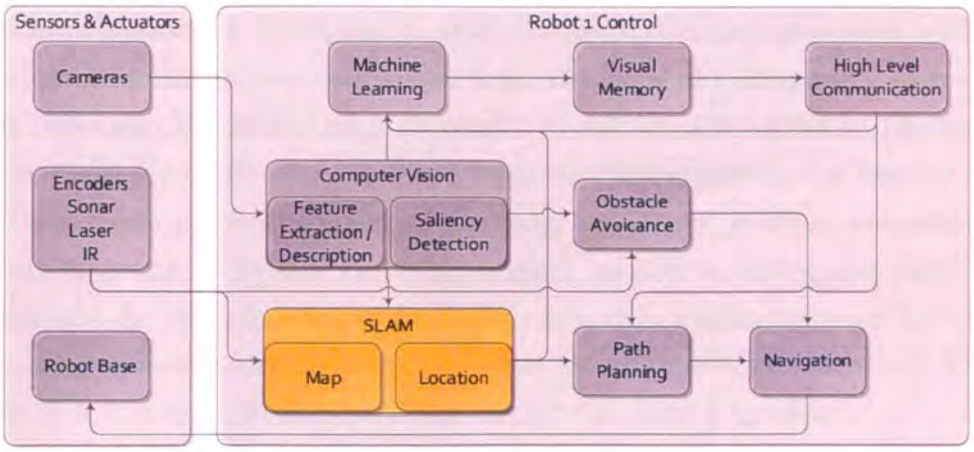


Figure 2.5: SLAM

- The estimation of the trajectory of the robot.
- The estimation of the map by means of a series of measurements.

The main problem raised with SLAM comes from the uncertainty of the measurements, due to the sensory noise or technical limitations. Probabilistic models are widely used to reduce the inherent errors and provide satisfying estimations [68]. SLAM is considered to be a complex task due to the mutual dependency between the map of the environment and the pose of the robot. This means that, if an error is made in the estimation of the pose, this would induce an error in the estimation of the map and vice versa. Smith et al. introduced the Kalman Filter to the field of robotic map-building [69]. The Kalman filter only handles linear systems, however for nonlinear systems the Extended Kalman Filter (EKF) can be used. The two key computational solutions to the SLAM problem are through the use of the Extended Kalman filter (EKF-SLAM) and through the use of Rao-Blackwellized particle filter (FastSLAM) [60]. The Kalman filter reduces the computation load by the assumption that the sensor noise follows a Gaussian distribution. Instead of maintaining the transition among all the states, only the Gaussian distribution parameters are maintained for each state. The particle filter reduces the computation load by reducing the number of samples - through iterative resample process; the particles converge at the real robot location. A comparison between these two methods is provided in [70]. Thrun et al. [71] also proposed a method using sparse extended information filtering (SEIF) and state that it compares well with the Extended Kalman Filter. Also, Chen et al. [52] identify the inherent relationship between the state estimation via the Kalman Filter versus Particle Filter and Expectation Maximisation techniques, all of which are derivations of Bayesian techniques and make use of Bayesian rules to correct posterior information by previous actions and perceptions. Chen et al [52] also cover the non-probabilistic approaches not covered in other surveys. More recently, methods relying on pose graphs to model and improve the estimations were also presented in literature [68].

Data association is a fundamental part of the SLAM process, since wrong data associations will produce incorrect maps. Loop closing and kidnapped / wakeup robot problems must also be tackled for a successful SLAM implementation. The term 'kidnapped' is used when the robot is lifted from its current (generally known) place in the map and taken to another part of the map. 'Wakeup' is when the robot is first started. In the case of kidnap the robot should be able to re-localise itself. In the case of wakeup the robot needs to figure out where it is within the map. Loop closure occurs during exploration and is when the robot revisits the same place [68]. The robot should be able to recognise the place and amend the map if necessary.

Multi-robot SLAM

Özkucur et al. [72] extended the map building onto a multi-robot platform where robots can observe each other and non-unique landmarks using visual sensors and merge maps by propagating uncertainty. Gil et al. [60] state that the SLAM problem becomes harder when more robots participate in the construction of the map, since the dimension of the state increases and indicate that multi-robot SLAM approaches can be grouped in one of the two following solutions:

- Approaches in which each robot estimates its own individual map using its observations. At a later stage, a common map is formed by fusing the individual maps of the robot team.
- Approaches where the estimation of all the trajectories and the map is made jointly. A single map is computed simultaneously using the observations of all the robots.

Visual SLAM

Mapping alone does not give information about the environment itself. This is especially so when maps are built on data provided by laser and sonar scanners combined with odometry. Most of robotic mapping is performed using sensors that offer only a 2D cross section of the environment around them, the most common being laser scanners. With the introduction of digital cameras, this later evolved into Visual SLAM whereby more knowledge about the environment could be obtained and included within the map. In the past decade various attempts were carried out to create a SLAM system just based on camera vision, some of which were very successful. Structure from motion in computer vision and SLAM for mobile robots could be considered as two views of the same problem. Most of the approaches to visual SLAM (VSLAM) are feature-based where a set of significant points in the environment are used as landmarks [60]. When the robot observes a visual landmark in the environment, it obtains a distance measurement and computes a visual descriptor as will be further discussed in Section A.3.4.

The VSLAM process can be described as estimating the poses of the camera from its data stream (video and depth), in order to reconstruct the entire environment while the camera is moving [68]. Some of the most common methods for calculating depth from vision sensors are discussed in Section 2.2.2. Influential works on VSLAM include the work by Davison et al. [73] in MonoSLAM using an Extended Kalman Filter, FastSLAM by Montemerlo et al. [74, 75] using Rao-Blackwellized particle filtering, RatSLAM [76] by Milford et al. which is based on biologically inspired visual SLAM system modelling a rodents brain and PTAM[77] / PTAMM [78] by Klein, Castle et al. In PTAM the aim is to track a calibrated hand-held camera in a previously unknown scene without any known objects or initialisation target, while building a map of this environment. Klein at al.[77] state that the two most convincing systems for tracking-while mapping a single hand-held camera are those of Davison et al. [73] and Eade and Drummond [79, 80]. Both systems can be seen as adaptations of algorithms developed for SLAM in the robotics domain (respectively these are EKF-SLAM [81] and FastSLAM 2.0 [75] and both are incremental mapping methods: tracking and mapping are intimately linked, so current camera pose and the position of every landmark are updated together at every single video frame. Klein at al.[77] argue that although the work by Davsion et al. [73] and Eade and Drummond [79] go to great lengths to avoid data association errors they still do not achieve good enough robustness. These two SLAM methods start with covariance-driven gating ("active search"), and then further perform binary inlier/outlier rejection with Joint Compatibility Branch and Bound (JCCB) [82] (in the case of [73]) or Random Sample Consensus (RANSAC) [83] (in the case of [79]). This concern motivated a split between tracking and mapping in [77]. If these two processes are separated, tracking is no longer probabilistically slaved to the map-making procedure, and any robust tracing method desired can be used. Whereas the work in [73] is limited by the frame-to-frame scalability of incremental mapping approaches which mandate "a sparse map of high quality features", Klein et al. [77] implement the alternative approach, using a far denser map of lower-quality features. In PTAM however, the scene should be mostly static and the user will spend most of his/her time in the same place. Exploratory tasks such as running around a city are not supported. As already highlighted in Section 2.2.2, there are two major problems to be solved in order to exploit the richness of vision for robot SLAM: the feature recognition and tracking problem, and the 3D reconstruction problem [52, 84]. Feature tracking is the problem of estimating the locations of features in an image sequence , details of which can be found in Appendix A in Section A.3.4. Gil et al. [60] state that the case of visual SLAM is particularly difficult since:

- The landmarks cannot always be detected from different viewpoints. As a consequence, it is difficult to re-detect previously mapped landmarks.

- The description of the points must be invariant to changes in viewing distance (scale) and viewing angle.

Two steps must be distinguished in the selection of visual landmarks:

- The detection of interest points in the images that can be used as reliable landmarks. The points should be detected at different distances and viewing angles, since they will be observed by the robot from different poses in the environment.
- The interest points are described by a feature vector which is computed using local image information. This descriptor is used in the data association problem and decides whether the current observation corresponds to one of the landmarks in the map or is a new one.

2.4 Machine learning and robot memories

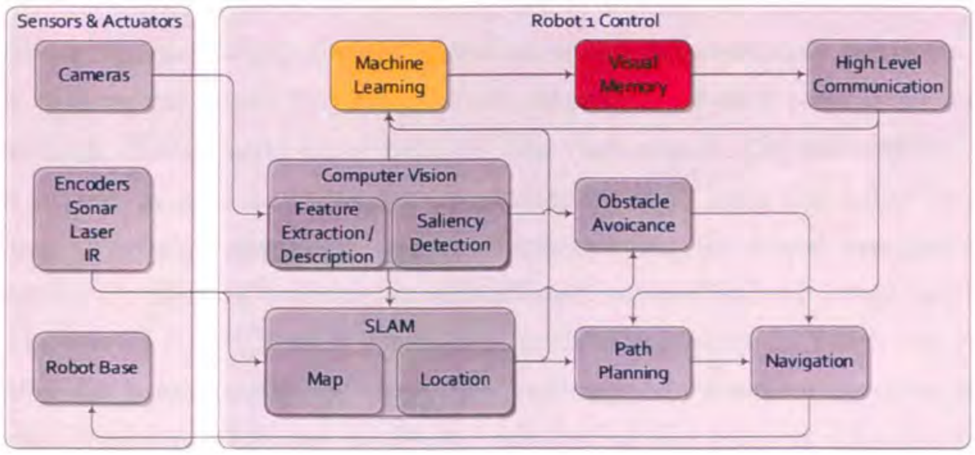


Figure 2.6: Machine learning and visual memories

Machine learning and robot memories go hand in hand (see Figure 2.6). Within the scope of this thesis, robot memories are restricted to mainly visual memory, which is the ability to recall previously learnt visual information. Machine learning is a scientific discipline within the artificial intelligence domain which takes input empirical data, such as that from sensors or databases to yield information by extracting rules or patterns from that data [46]. Various types of machine learning methods are available and the method used highly depends on the application and data availability. Some of the most common methods include supervised, unsupervised, semi-supervised and online learning. In addition to these kinds of learning, there are others, such as reinforcement learning whereby the learning method interacts with its environment by producing actions that result in rewards or punishments. These learning methods are distinguished by what kind of feedback the critic provides to the learner. In supervised learning, the correct output is provided whereas in unsupervised learning, no feedback is provided at all. In reinforcement learning a quality assessment is provided in the form of a reward/punishment based on the learner's output [85]. More literature about the various learning methods is provided in Appendix Section A.4.

In robotics, the most common use of visual memory involves a robotic agent which is first taught a route and then has to localize itself within the environment and re-track the path followed previously by recalling what memorised earlier [86, 87, 88, 89, 90, 91, 92, 65]. A similarity score between the view acquired by the camera and the database images is used as input for the controller that leads the robot to its final destination [92]. The most widespread approaches to visual navigation are the model-based, and the appearance-based approaches [92]. Model-based approaches rely on the knowledge of a 3D model of the navigation space. The model utilizes perceived features (e.g., lines, planes, or points), and a learning step can be used for estimating it. Conversely, the appearance-based approach does not require a 3D model of the environment, and works directly in the sensor space. The environment is described by a topological graph, where each node corresponds to the description of a position, and a link between two nodes defines the possibility for the robot to move autonomously between the two positions. Cherubini et al. [92] compare six appearance-based controllers relying on visual memory. A similarity score between the view acquired by the camera and the database images is used as input for the controller that leads the robot to its final destination. Pattern recognition and machine learning for vision are important in this context [52]. During learning in vision-based navigation, not every single image needs to be stored [65]. There are scenarios, such as corridors, in which the views are very similar for a long period of time. Those images do not provide data useful for navigation. Therefore, they can be filtered out during the learning stage, so that only images which are sufficiently different from their predecessors must be stored. Mendes et al. [65] used the Sparse Distributed Memory(SDM) to carry out such task. in SDM a new image is only stored if there is no image within a predefined radius of the SDM. New images that are less than the activation radius from an already stored image are most probably unnecessary and can be discarded with no risk of impairing the performance of the system. Visual memories will be studied in more detail in Chapter 4, where the contribution towards bettering the state of the art is also provided.

2.5 Multi-robot systems (MRS)

With the basic problems concerning single-robot sensing and control being fairly well understood, mobile robot deployment scenarios are rapidly evolving from single robot to multiple robot systems (MRS) within a cooperative paradigm [41]. MRS can accomplish tasks that no single robot can accomplish by itself, since ultimately a single robot,

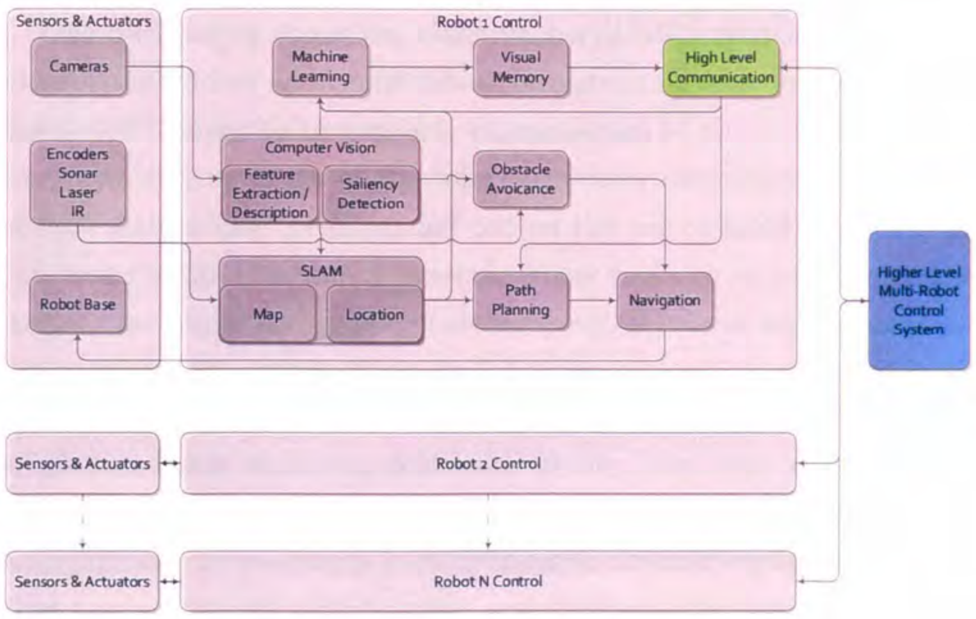


Figure 2.7: Team Collaboration

no matter how capable, is spatially limited [93]. MRS provide the advantage of having many robots being distributed within the available space to carry out diverse tasks at the same time. Ideally, as seen in Figure 2.7, such tasks are carried out by robots collaborating and working together as a team in order to achieve the required goal. A key driving force in the development of cooperative autonomous MRS is their potential for reducing the need for human presence in dangerous applications. To mention some examples, such tasks could include search and rescue or surveying and patrolling of large areas. There are still many open issues in MRS some of which include 1) swarm control and fault tolerance, 2) authority delegation and action selection, 3) communication structure, 4) heterogeneity versus homogeneity of robots, 5) resolution of conflicts and other related issues [94]. Some possible solutions are discussed next.

A MRS generally starts with a collective of homogeneous or heterogeneous set of mobile robots which are generally expected to operate in both open unstructured landscapes and structured outdoor and indoor environments [95]. Collective behaviour denotes any behaviour of agents in a system having more than one agent [93]. This generally introduces redundancy and one therefore expects the collective to be more fault-tolerant when compared to its single robot counterpart. As will be discussed in Section 2.5.1, another possible advantage comes with the use of robot teams capable of merging overlapping information and thus compensating for sensor uncertainty [96]. A MRS is not just the extension of a single robot by putting in more robots in the environment and it cannot be simply regarded as a generalization of the single robot case. When using a MRS, each robot may be designed for a different task, and the required goal is achieved with proper collaboration by the robots. This would provide a more generic structure as the robots would be able to reconfigure themselves as required. One question is about the effectiveness of MRS over single-robot versions, and to what extent adding additional robots brings diminishing returns. The proposed approaches for MRS need to be precisely characterized in terms of assumptions about the environment and in terms of the internal system organization [97]. In the last decade several studies have been carried out on the use of MRS including strategies, formations, task distribution and delegation. There was also an increase in complexity, namely larger team sizes and greater heterogeneity of robots and tasks [98]. Several studies on coordination architectures and demonstrations of coordinated behaviour in multi-robot systems have been carried out. Most of these systems however lack a theoretical foundation that can explain or predict the behaviour of a multi-robot system. Gerkey et al. [98] provide a candidate framework for studying such systems.

Due to significant achievements in MRS research one now expects to see increasingly larger robot teams engaged in concurrent and diverse tasks over extended periods of time rather than just having multiple robots observing similar targets or large robot groups flocking together [98]. Robots are now expected to be reliable and robust, able to improve their capabilities by learning from their environment through interaction with people and other robots, adaptable and autonomous in a dynamic unstructured environments, modular, collaborative with other robots and humans such that they are able to tackle problems and carry out tasks which otherwise they would not be able to carry out by themselves and also able to evolve, capable to elaborate strategies according to what the environment necessitates [36]. A MRS should be a solid network of robots capable to interact successfully and share information.

Cooperating robots have the potential to accomplish a single task faster than a single robot. Having said that however, due to the extent and dynamics of the environment it is difficult to achieve efficiency and applicability for general-purpose cooperative robot systems. When a MRS is being designed for a particular task or set of tasks, several factors need to be considered. One has to decide, given on the environment in which the multi-robot system will operate, if the system will consist of homogeneous or heterogeneous robots, have a centralized or distributed organization, the tasks to be performed will be loosely or tightly coupled together and if the robots are to work on various tasks in an ensemble rather than individually. These would generally define the multi-robot team architecture which will be further discussed in Section 2.5.2. Also, there are other robot issues one has to consider such as how communication between robots will be carried out, how the tasks will be assigned and if the robots will have their own specialisation. For a robot to be able to work in a MRS one would typically incorporate wireless Ethernet as the basis for communication, vision based sensing amongst other input sensors and an on-board computer, hence having the ability to support a significant level of autonomy as well as robot-robot and human-robot cooperation [95]. Moreover, given that nothing is ideal in the real world, one has to expect and compensate for uncertainty in measurement reading from sensor and resulting action from actuators. MRS need to take into account the uncertainty, the limitations, and the mistakes arising from the processing of sensor information [97].

In recent years, research on intelligent mobile robots for area monitoring is taking place and applied for area coverage [99, 100]. Folgado [101] highlights some video-surveillance projects such as: VSAM, CAVIAR and a large number of projects related to traffic surveillance, among others. Cooperative multi-robot approaches have been proposed for surveillance [102, 103, 104]. MRS could contribute to solving problems especially when it comes to robot positioning. Without a priori knowledge, the uncertainties in the environment increase the complexity of system design. Best configurations have to be analysed and cases such as ways to avoid occlusion could be identified. This would require the robots to be able to localize themselves and navigate within the unknown space. There is a lot of research interest in the study of robot learning that can let robots adapt to the environment and other variables to accomplish surveillance tasks. Moreover studies on how robots can cover each others blind spots need to be investigated. Furthermore, the system should be reliable enough to continue working even if parts of the system, e.g., some robots, fail in functioning. One such solution is provided by Feng et al. [105]. Several methods are also available for exploration and target searching including, potential field-based exploration [106], swarm intelligence exploration [107], landmark-based exploration [108] and hop-count gradient-oriented searching [109]. Various multi-robot formations are possible, each suited best for the particular application. Some possible formations include: swarms, colonies or simply robot-collectives [97].

A lot of effort is currently being put by researchers on RoboCup[7]. The main focus of RoboCup is Robotic Soccer (RoboCup Soccer), although other application domains exist focusing on different scopes like disaster rescue (RoboCup Rescue), robotics education for young students (RoboCupJunior) and human assistance on everyday life tasks (Robocup@Home) [110]. In RoboCup Soccer the aim is that of having a team consisting of generally homogeneous robots to interact together in a constructive way in order to play a game of soccer and scoring by shooting a ball in the opponents goal. This environment is considered as hostile due to the opponents team interfering actions. Kiener and Styrk [111] provide a case study of a strongly heterogeneous autonomous robot team composed of a highly articulated humanoid robot and a wheeled robot with the main task being that of finding and following a ball and finally kick the ball into the goal. Based in the RoboCup scenario Pagello [112] provides an omnidirectional distributed vision system (ODVS) where they studied ways to enhance the cooperative capabilities of the robots allowing them to track moving objects in a highly dynamic environment by sharing the information gathered by every single robot.

[7]http://www.robocup.org/

In [85], Panait and Luke survey a large spectrum of areas useful for multi-agent systems, including reinforcement learning, evolutionary computation, game theory, complex systems, agent modelling and robotics. They mostly focus on cooperative multi-agent learning, where multiple agents are cooperating rather than competing with one another to solve a joint task or to maximise utility. Buşoniu et al. [113] state that significant progress in the field of multiagent learning can be achieved by a more intensive cross fertilization between the fields of machine learning, game theory, and control theory. Learning involves interaction amongst team members, with others external to the team, and with the environment. Exposure to individuals with different expertise and experience is a vital source of team learning. Interaction with dissimilar others promotes learning by exposing actors to new paradigms and by enabling the cross-fertilization of ideas. Team learning will be further discussed in Section 2.5.3.

Most of MRS systems so far have been tested on simulation environments and is only lately that the shift from simulated to real environments can be seen. Trying to simulate a real environment has its own limitations. A robot will only be able to replicate what was observed in simulations within the real environment only as much as the simulation can faithfully replicate a real environment. Multiple-robot systems have implicit "real-world" environment constraints, which are more difficult to model and reason about than traditional components of distributed system environments [93]. Real environments using real robots pose bigger challenges than their simulated counterparts as some factors such as background and sensor noise, unexpected dynamic movement and conditions within the environment cannot be filtered out or ignored as is done in simulations. The acquisition of knowledge from the environment, makes it more challenging to build actual experimentation settings for MRS [97]. Mckee and Varghese [95] highlight four issues which need to be considered when working with multiple robots in a real environment. One has to consider whether the team is composed of homogeneous or heterogeneous robots, what happens in case of partial or total failure of a particular robot, the coordination required for a successful completion of a task and what interfaces are used between the robots. Apart from motion drift errors which were already present in real robots, in a multi-robot scenario new issues come up, one being interference between robots. If the robots are homogeneous or use similar active sensors, such as laser, Kinect and sonars the overall performance can be reduced due to cross-talk / interference between the sensors and worse still data acquired by one robot could be corrupted by another robot in the same environment. Also, the more robots are used the longer detours may be necessary in order to avoid collisions with other members of the team [96]. In practice, one also has to deal with a limited communication range between robots. This could either happen due to large distances between robots or by temporal network errors. Measures have to be taken to bridge large distances between robots to maintain communication over the wireless network.

Robots have to be able to work independently when such communication interruption occurs and devise ways to maintain optimality as much as possible. In the case of multi-robot exploration robots might end up revisiting areas previously explored by other robots leading to suboptimal behaviour [96]. Robot communication, networking, cooperation and collaboration are further discussed in Appendix A, Section A.6.

2.5.1 Robot teams

A team is a group of entities with a full set of complementary skills required to complete a task. A task, is a subgoal that is necessary to achieve the overall goal of the system, and that can be achieved independently of other subgoals [98]. Robot teams are very useful when it comes to carry out collaborative tasks which otherwise would have been infeasible for a single robot. Some such situations include the exploration of hazardous environments, the moving of heavy or large objects and the effective patrolling of an area. Robot teams are also expected to complete a task more rapidly, execute tasks beyond the limits of a single robot and gain better performance by using multiple specialised robots rather than one superbot. Robot teams also have the advantage of having highly distributed sensors. The use of robot teams introduces new challenges. In particular a critical one is how to best take advantage from the variety of available resources [114]. An optimal performing team is one which is able to take advantage of the strong points of each available robot. This is even more strongly emphasised in a heterogeneous environment where robots are not the same. Various types of robots, some of which are discussed in Appendix Section A.1, can be used in teams as long as they are able to communicate. These could be of different size, different construction, and also have different onboard sensors and actuators. Burgard et al. [96] provide a solution for using teams of heterogeneous robots in a real-world scenario.

Mckee and Varghese [95] propose a multi-team model for MRS, whereby multiple subsets of robots are drawn from a larger pool to form multiple teams. Each team has an assigned task that is to be distributed among the members of the team. Nouyan et al. [115] present a self-organized system of robots that displays a dynamical hierarchy of teamwork (with cooperation also occurring among higher order entities). Their study shows that teamwork requires neither individual recognition nor differences between individuals. Bradshaw et al. [116] discuss some of the challenges and requirements for successful coordination in a multi-team human-robot field exercise. Hoffman and Breazeal [117] propose an adaptive action selection mechanism for a robotic teammate to a human, making anticipatory decisions based on the confidence of their validity and their relative risk thus expecting an improvement in task efficiency and fluency compared to a purely reactive process.

Sharing knowledge between robots requires methods to effectively encode, exchange and reuse data. Robots are also expected to interact with other smart devices such as intelligent surveillance cameras, tablets and mobile devices. Therefore, Machine-to-machine (M2M) wireless communications is becoming more important than the current paradigm focusing on machine-to-human or human-to-human information exchange [118]. Comparatively little research has addressed the sharing and reuse of knowledge [28]. Some researchers have proposed sharing pooled data using an Internet search engine or a cloud computing framework. Others have suggested embedding knowledge directly into objects. Attempts such as the planning domain definition language target the standardisation of plan languages and planning domain specifications. Another approach aims at creating abstract representations for high-level knowledge that can be shared across multiple platforms. An ambitious project called RoboEarth [28] aims at collecting, storing and sharing data between robots, independent of their specific hardware. Moreover data is stored in a form of linked database where computer-aided design (CAD) models, their semantic descriptors and other properties and relations to other objects can be stored. This might also include other robot instructions for handling such object.

2.5.2 Multi-robot architecture and team structures

For successful team operations it is useful to have architectures in which individual robots are aware of being part of a team and act accordingly as team players [97]. Moreover, when designing multi-robot architectures and setting up team structures, one tries to maximise one's resources. Robots have to be allocated accordingly, depending on the task at hand and decisions need to be taken on which robot should execute which task [119, 98]. Working with multi-robot teams also creates challenges in command and control, whether top-down, bottom-up, or a combination of these [95]. These must be reflected in the capabilities incorporated in the robot architecture itself and in the global commands that need to be translated into actions for individual robots. Architectures proposed for MRS tend to focus on providing a specific type of capability to the coordinating robot team [94].

As already discussed in Section 2.1.2 researches were inspired by living organisms to create their multi-robot architectures mimicking biological systems. Other researchers however, found inspiration from other sources creating complex models arising from cognitive science and economics [97]. Gerkey et al. [98] showed how Multi-Robot Task Allocation (MRTA) problems can be studied in a formal manner by adapting to robotics some of the theory developed in relevant disciplines that study organizational and optimization problems. These disciplines included operations research, economics, scheduling, network flows, and combinatorial optimization [98]. Vig and Adams [120] presented a market-based task allocation procedure based on robot bidding.

There are several properties that are desirable in the allocation mechanism of a robotic team and which should be considered when designing such architectures such as task planning, mathematical soundness, distributabiliy, decentralisation, scalability, fault tolerance, flexibility, adaptability, responsiveness, swarm control, human design of mission plans and role assignment [97, 94, 114]. One also has to look at execution models, constraints, optimisation and computational complexity [114]. Adaptivity refers to the ability of the MRS to modify its own behaviour over time, depending on changes due to the dynamic environment, changes in the system mission or changes in the system composition or capabilities, so that the performance of the entire system can either be improved or at least not degraded. Fault tolerance is the ability of the MRS to deal with individual robot failures or communication failures, between two or more robotic agents, that may occur at any time during the mission. MRS which present both the above features can be said to be robust, where robustness is, therefore, the ability of the MRS to be both adaptive and fault tolerant [97].

To facilitate robust and speedy deployment of robot teams, teamwork architectures are increasingly used to automate the interactions between team-members, such as synchronized task execution and task allocation. This allows the designer to focus on developing the task work, rather than the teamwork [121]. Most of the work in MRS has been devoted to the definition of different architectures and several architectures were developed being either 1) *behaviour-based architectures* ruling the interaction between the behaviours of individual robots [42], 2) *Sense-Model-Plan-Act architectures* where the system develops an internal model upon which to conduct its planner [122] or 3) *hybrid architectures* which are a combination of both of the above methods [123]. The adopted control architecture for many robot systems, either alone or working in a team is a hybrid comprising deliberative and behavioural components with the balance between the two determined by the task performed and the scale and number of robot systems [97]. Good team architectural design allows for a pool of robots to form into a set of teams and a team can form into sub-teams thus tasks and roles are not only assigned to individual robots but also to teams [95]. Robot architectures can be generally classified as either being a centralised architecture or otherwise a distributed architecture [97]. When centralised robots follow a hierarchical order and the system is organised by having a robotic agent acting as a leader that is in charge of organising the work of the other robots. In centralized architecture, one central control unit manages whole information about environment and subordinate robots, decomposes and assigns tasks through plan algorithm and optimize algorithm, organizes robots to complete tasks by sending commands. The leader is involved in the decisional process for the whole team, while the other members act according to the directions of the leader. Elements on the lowest-level are ruled by a few elements on higher levels. The organization and distribution of complexity looks like a military pyramid: high complexity below and a low complexity on the top. This system is sometimes also referred to as having a 'vertical operational principle'. This model has disadvantages in flexibility, integrity, expandability and fault tolerance [94]. On the other hand, when distributed, the system is composed of robotic agents which are completely autonomous in the decisional process with respect to each other; in this class of systems a leader does not exist. Each individual operates on local information. The global goals would either be implicit or known. The control rules may be fuzzy and the complexity is similar on all levels of abstraction. This configuration can sometimes be also be referred to as following the 'horizontal operational principle' [36].

2.5.3 Team learning

Several studies on team learning within multi-robot teams have been carried out [124, 125, 126, 127, 128, 129, 130, 85]. Team learning may be divided into two categories: homogeneous and heterogeneous team learning [85]. In homogeneous team learning, all robots are assigned identical behaviours, even though they may not be identical (e.g. different construction or different processing power). Homogeneous learners develop a single robot behaviour which is used by every robot on the team. On the other hand, in heterogenous team learning, the team is composed of robots with different behaviours, with single learner trying to improve the team as a whole. Heterogeneous team learners can develop a unique behaviour for each robot with the bulk of research focused on the requirement for the emergence of specialists, generally providing better solutions through robot specialisation. Agents can act heterogeneously even in a homogeneous team learning environment. This happens when the homogeneous behaviour specifies sub behaviours that are different based on the robot's initial condition or its relationship with other agents. As will be seen in Section 5.2.2 this approach was used by having a homogeneous set of robots learning the environment with one robot having different initial conditions and set up as being visually impaired accepting input only from the other robots and not from the environment itself.

Yang and Gu [131] provide a survey on multi-agent reinforcement learning for multi-robot systems. As highlighted in both [131] and [85] scalability is a problem for many learning techniques but especially so for multi-agent learning. Yang and Gu [131] state that there is a lack of theoretical grounds which can be used for proving the convergence and predicting performance. Panait and Luke [85] make a bold statement and say that one cannot learn the entire joint behaviour of a large, heterogeneous, strongly inter-communicating multi-agent system. In [113] Buşoniu et al. discuss in detail several multi-agent reinforcement learning techniques for fully cooperative, fully competitive, and mixed tasks. When it comes to multi-robot scenarios, competitive behaviour would generally consist in pursuit-evasion or one-on-one competitive games [93]. Buşoniu et al. [113] state that control theory can contribute in addressing issues such as stability of learning dynamics and robustness against uncertainty in observations or the other agents' dynamics. The most common alternative to team learning in cooperative multi-robot systems is concurrent learning, where multiple learning processes attempt to improve parts of the team [85]. Typically each robot has its own unique learning process to modify its behaviour. When applying single-robot learning to stationary environments, the agent experiments with different behaviours until hopefully discovering a globally optimal behaviour. In dynamic environments however, the robot may at best try to keep up with the changes on the environment and constantly track the shifting optimal behaviour. Things are even more complicated in multi-robot systems, where the agents may adaptively change each others' learning environments.

2.6 Summary

This chapter analysed in detail various areas related to multi-robot vision. The suggested multi-robot platform was dissected into six sections and the state of the art for each of these sections was presented. This chapter looked at various methods available in the literature which tackle problems related to human-robot collaboration and how some of the authors were inspired by biological systems. Various areas in the robotics field related to navigation, path planning, mapping and exploration were presented. Computer vision, with special emphasis on object detection methods, depth calculation and 3D measurements were discussed. Machine learning techniques and how they can be used to create robot memories were also discussed. Finally methods on how all these different areas can be combined and extended onto multi-robot platforms were presented.

Over the coming chapters, the above methods are studied, adapted and improved in order to have a multi-robot platform successfully operating in a heterogenous environment, whilst learning, sharing and fusing salient information about the environment between mobile robots, fixed cameras and humans.

"Science is nothing but developed perception, interpreted intent, common sense rounded out, and minutely articulated."

George Santayana, *Spanish Philosopher (1863 - 1952)*

Chapter highlights:
This chapter discusses a method which makes robots aware of their dynamic environment and identify both known objects and other yet unknown salient objects by using a mix of top-down and bottom-up approaches.

3 Robot Perception

Focusing mainly on robot vision, this chapter discusses methods on how robots are made aware of their dynamic environment and solutions are provided for integrating some methods related to areas of robotics, computer vision and machine learning. In the coming sections, the integration of various methods described in Chapter 2 were implemented and improved where necessary so as to achieve robots which are capable of sensing and processing visual input. Some of the covered methods include the comparison of images, the detection of areas which are of interest within the environment and also the capability for robots to identify specific objects. This chapter discusses a method which allows robots to find and identify both known objects and other yet unknown salient objects which might of interest. A mix of top-down and bottom-up approaches is suggested. The same methods could be used to support persons in an assisted living environment to identify objects and receive information about the object. Part of the task was therefore to design tags containing patterns which are both salient to humans and robots [22]. These tags, further described in Section 3.2.1, consist of a set of five coloured circles on a black rectangular background. A study was carried out to identify what people identify as salient within an image with the aim to replicate that onto our robots using efficient methods capable of operating in real-time. Section 3.1 looks at perception, visual attention and saliency and how these can be applied to robots using a bottom-up approach. Section 3.2 focuses on top-down and analysis on visual tag detection performance is carried out.

3.1 Perception, visual attention and saliency

Perception is the ability to see, hear, or become aware of something through the senses. The human visual system is extremely efficient at dealing with vast amounts of information. Humans can voluntarily focus their visual attention on a specific location, or be involuntarily attracted by things of visual interest. The saliency of an item is the state or quality by which it stands out relative to its neighbours. Visual attention has been studied in great depth, by vision scientists and physicians, and more recently by computer vision and robotics researchers [132, 133, 134, 135, 136].

In the domain of computer vision, efforts have been made to model the mechanism of human attention, especially the bottom-up attentional mechanism. Such a process is also called visual saliency detection. Saliency is used for object/person segmentation, object/person recognition and compression of visual data [137] and for image retrieval [138]. Chang et al. [139] also suggest the use of gist by capturing holistic characteristics and layout of a scene together with saliency to be applied for robot vision.

A pilot study to identify points of general interest identified by humans on images using an eye-tracker was carried out. This was done in collaboration with other researchers[1] forming part of the Robot Vision Team (RoViT) at Kingston University. A saliency model, further described in Section 3.1.1 was also developed [24] to emulate the human visual interest on a robot. This model combines both the bottom-up visual attention approach and the top-down task-oriented approach, both important for the robot application described earlier in Section 1.2.1.

Thirteen subjects between the ages of 20 and 45 participated in this pilot study. It was assumed that all visual functions were sound for all subjects. A set of 250 images was generated. This consisted of a mix of images taken from the internet and others taken from the robot camera. These images, which were expected to contain items which humans would consider as salient, were then displayed on a computer screen. Each subject was asked to look at each image for five seconds with a one second break between each image. The subjects were also instructed not to look at anything specific. After fifty images a blank black screen was displayed for five seconds. This resulted in a database of 250 heatmap images highlighting the general regions of visual interest of our subjects. Figure 3.1 (left) shows the original views from a robot camera working in a mock environment with placed tags [22](Section 3.2.1) and pictures of faces [140]. A tag and a face can be seen on the right hand side of Figure 3.1 (top-left). The human fixation points captured using an eye-tracker are indicated in Figure 3.1 (right) where the red blobs represent the points of general interest of all thirteen people carrying out the test. The white regions within the red blobs indicate the strongest fixation points. Looking at Figure 3.1 (right), one can note that a human subject automatically looks at

[1]Dr Maria Valera Espina and James Cope

both tags and faces and also at any other prominent characteristic of the environment, all having strong fixation points on them.

Visually impaired persons might find it hard to quickly identify specific objects, depending on their impairment. In computer vision, a well-trained object detector can perform such tasks reliably and in real-time and can be used to help persons needing assistance. The object detector however will only work for known objects and, without user intervention, it is unable to learn any new interesting objects within its environment.

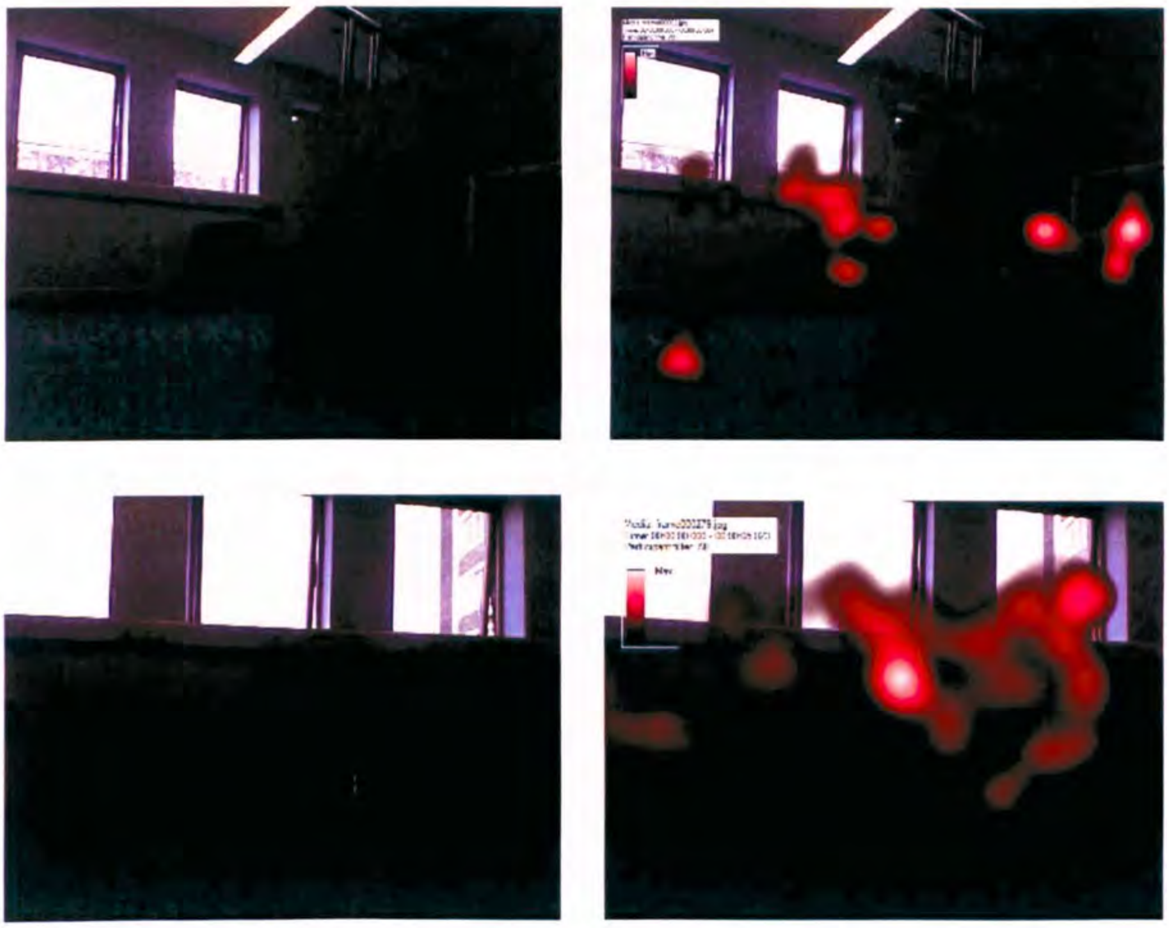


Figure 3.1: (left) Original View from robot camera, (right) Fixation points

Two different types of attention processing can be evoked when looking for objects of interest. If the search is voluntary and knowledge and goal oriented (i.e. specifically looking for something such as a face or specific object) then it will be a top-down mechanism. In this case the guiding mechanism is generally memory-dependent and uses anticipatory mechanisms to look ahead on where to look. This is the case with tracking moving objects. On the other hand if the attention is involuntary and triggered only by sensitivity to salient stimuli then it is said to be a bottom-up approach. In this case the bottom-up approach is considered to be memory-free and reactive [141, 142]. A number of saliency models have been proposed in the literature [132, 137, 143, 144, 145, 146, 147, 148]. The bottom-up approach is the most common approach to saliency map creation however a hybrid method combining bottom-up and top-down has been proposed by [143, 146, 147, 149]. Saliency maps have been used in rapid scene analysis [148], video surveillance [150] and compression for data transmission [149]. The features often used to extract saliency are based on gradient, orientation and colour information [132, 151, 148, 146, 147, 143, 152, 144, 150]. Frequency domain analysis [145] and wavelets [152, 150] were investigated as a means to extract the features needed to obtain the saliency maps. Su and Takahashi [153] introduced the concept of an importance map which is a scalar value image that topographically represents the perceptual importance of a visual scene. Saliency maps were applied in image region segmentation, object detection and robot vision simulation [153]. Lin et al. [154] introduce a new computational visual-attention model for static and dynamic saliency maps and improve over Itti's model of visual attention [132]. Lin et al. suggest the use of the earth mover's distance (EMD) to measure the centre-surround difference in the receptive field instead of using the difference of Gaussians (DoG) filter adopted by Itti's model [132]. They also consider a nonlinear operation for combining features into a set of *super features* followed by a winner-take-all (WTA) mechanism. Lin et al. also construct dynamic saliency maps from an input video by extending the computation of the center-surround difference over the spatio-temporal receptive field (STRF).

Up until present, most of the saliency algorithms are time consuming and computationally intensive, therefore unsuitable for real time operation on a robot. Su and Takahashi [153] state that the low-resolution output and heavy computational complexity of the saliency map still remain as crucial limitations for a wider utilization of the saliency map. Butko et al. [155] provide a real-time solution for visual saliency by proposing a fast approximation to a Bayesian model by using a Difference of box filters (DoB) over the image intensity channel. They empirically evaluate the saliency model in the domain of controlling saccades of a camera in social robotics situations by orienting a camera as quickly as possible towards human faces. Chang et al. [139] present a vision based navigation and localisation system for robots based on gist and saliency.

All the above mentioned methods try to emulate the visual attention of humans. Typically these methods are evaluated on how well they predict the actual specific locations that humans have fixated in eye-tracking experiments where humans are only instructed to look at some image with no specific target [155]. Eye tracking is the process of measuring the point of gaze (fixation) and motion of an eye (saccade and smooth pursuit) relative to the head and is done using eye trackers. There are a number of methods for measuring eye movement. The most popular variant uses video images from which the eye position is extracted. As stated in [156] apart from eye movements there are other important aspects of attention which need to be considered when carrying out model evaluation. Some other metrics might include the accuracy in correctly reporting a change in an image or predicting what attention grabbing items one will remember. Borji and Itti [156] review, from a conceptual perspective, the basic concepts of attention implemented in nearly 65 models, presenting their taxonomy providing a critical comparison of approaches, their capabilities and shortcomings.

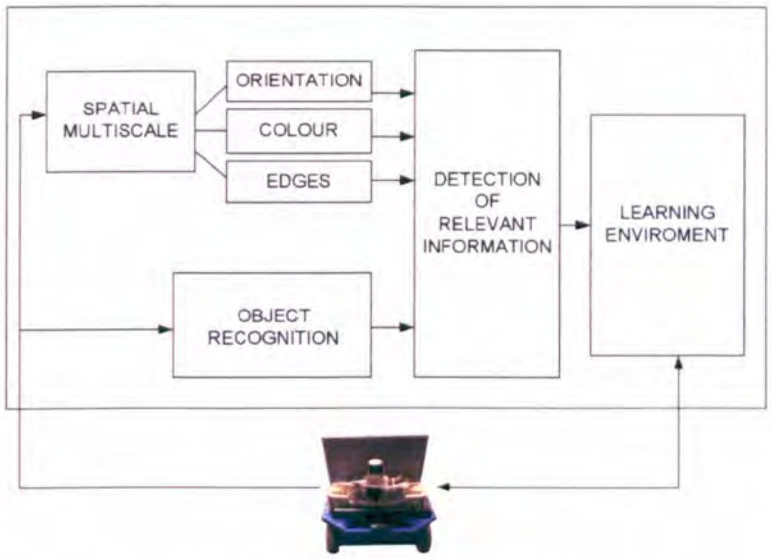


Figure 3.2: Robot Perception Model

A mix of top-down and bottom-up approaches were considered. Bottom-up approaches were used for the initial identification of salient areas in the region. Unsupervised machine learning and object detection, which are able to operate in real-time, were used for the rest. For the bottom-up approach, the computational model of visual attention called saliency maps formulated by Itti et al. [132] where the centre-surround mechanism is used to extract low-level features such as colour, intensity, and orientation that are different from surrounding areas was studied. For top-down, a way of combining machine learning with saliency in a way that the robot learns the more probable areas where interesting objects are likely to occur within a scene, was tested. These learnt areas would eventually becoming a visual memory (further discussed in Chapter 4). For other known objects, such as faces and tags boosted classifiers were used. These are further described in Section 2.2.1. Figure 3.2 illustrates the robot perception model used on the proposed platform. This model tackles the following limitations:

- employed robots have limited storage space and processing power;
- although the on-board pattern detector works in real-time, it will only successfully detect the objects for which it has been trained, (moreover, training is time consuming and needs a very large training set of labelled data);

- saliency algorithms can identify unknown regions of interest within a scene, however they are computationally intensive and unsuitable for real-time operation [153].

This allows to take advantage of the real-time capability of pattern detection, the ability to detect new salient areas and the ability to memorise scenes efficiently in order to have a robot capable of autonomously identifying known objects whilst also learning new environments. Although not limited, it is assumed that, once deployed, the robot will operate in the same environment for most of its lifetime; the longer the robot remains in the area the more refined its memory will become [23]. As the robot explores an area, looking for objects for which it has already been trained, the input video stream is temporarily recorded into a buffer. As soon as the robot becomes idle, the robot begins processing the captured video as a background task, looking for salient regions. If the portions of the video contain sufficiently salient information (when compared to what humans would generally find as being of interest) the keyframes will then be stored onto visual memory. This ensures that only regions of interest are stored. Saliency maps implement the extraction of low-level features such as colour, intensity, and orientation that are different from their surrounding areas [132]. The implementation is described in the coming sections.

3.1.1 Saliency model

Saliency was calculated for a set of uniformly spaced 25×25 windows within the image based on colour, entropy and orientations present within each window. Entropy filters are used to statistically measure the level of randomness and to detect areas with high rates of change. Furthermore Gabor filters are employed to detect edge orientation and the HCL (hue, chroma, lightness) colour space was used for extraction of colour content [24]. The saliency for each window is calculated from the similarities of these features to those for the other windows within the image. This is done at three different scales and then combined to achieve the final saliency. The eye-tracker pilot test indicates that humans tend to give more attention to the central region of the image rather than the lateral sides. To replicate this, a Gaussian weighting was thus used over the outcome, giving a lower weight to salient points which are further away from the image centre. A threshold was also used. This threshold was set based on experimental evidence, when compared to eye tracker ground truth data. If the resulting salient point is less than the threshold (low saliency), the pixel value is set to 0, otherwise its value is retained. Fig.3.3 shows the results of the algorithm. The left image contains the saliency obtained from the algorithms on the images provided in Fig.3.1. It must be noted that the face and tag have strong fixation points on them but also there is also fixation happening in the central region. This is captured using the centre region Gaussian filter. If the ratio value (saliency in centre/overall salience) is low, as in Fig.3.3 top right, then the frame is not fed into visual memory. If on the other hand the value is high, as in Fig.3.3 bottom right, the frame is fed into the visual memory. Once processing is completed, the recorded video stream can be discarded and a new video sequence is recorded for further training. The resulting visual memory is available for future use for environment matching and recognition or possibly for sharing between other robots as described in [25].

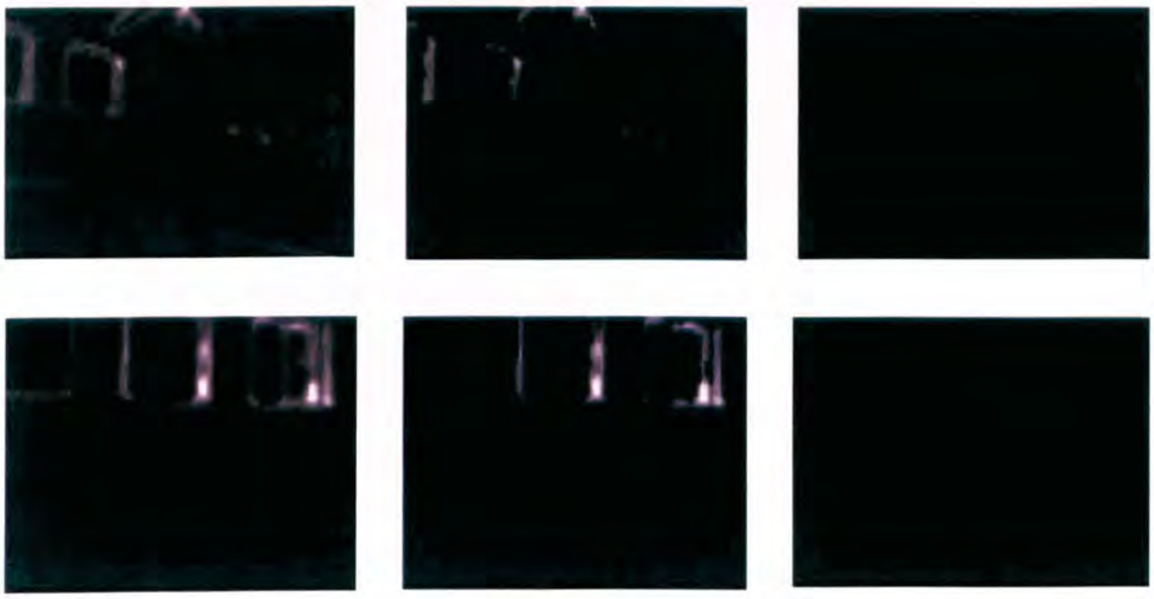


Figure 3.3: (left) Saliency outcome, (middle) Threshold pixels, (right) centre fixation

In the coming sections, the filters used to identify saliency areas within a frame and for object / tag identification as mentioned above and illustrated in Figure 3.2 will be described in more detail.

Entropy filter

In order to highlight image areas with high rates of change, an entropy filter as described in Appendix Section B.2 was used. A measure of the greyscale variation is produced. A histogram is calculated for the greyscale values of the pixels within an image patch, using eight equal-sized bins [24]. A measure of entropy for the patch is then calculated using Equation 3.1 where $p(x_i)$ is the fraction of the pixels whose greyscale values place them in bin i.

$$f_e(x) = -\sum_{i=0}^{8} p(x_i) \log p(x_i) \tag{3.1}$$

Areas with high greyscale variance (an even distribution of greyscale variance) will produce a high entropy value, whilst a uniform area will yield the minimum value. The entropy image is then normalized using Equation 3.2. To highlight big changes and eliminate small changes, different normalization factors were used as shown in Equation 3.2 where D is a decay rate and N is the number of pixels within a patch.

$$f_e(j) = \left(\frac{f_e(x)}{N}\right)^D \tag{3.2}$$

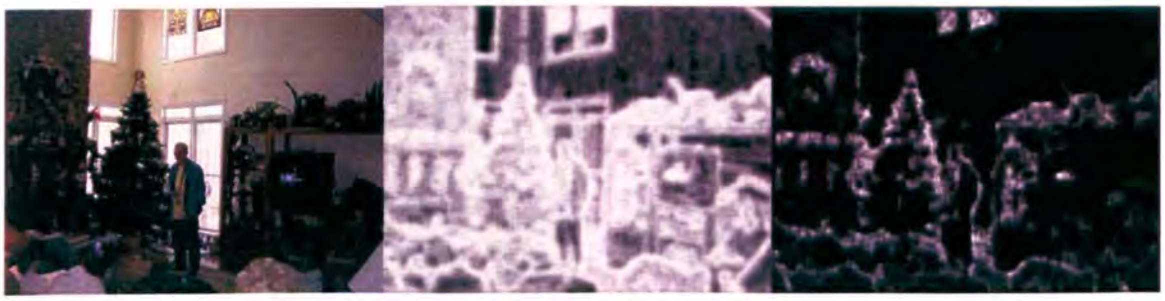


Figure 3.4: (left) Original Image, (center) Entropy Filter outcome when passed over grayscale image, (right) Normalised entropy image

The larger is the D exponent factor, the faster values close to 0 will decay when compared to those close to 1. E.g. $0.2^5 = 0.00032$ whereas $0.8^5 = 0.32768$. This faster decay can be seen in Figure 3.4 where D was set to 10.

Colour(I) - Probability of density colour filters

Intuitively, regions with colours which stand out are more likely to be salient. A colour segmentation technique based on assigning weighs to pixels in the scene (see bottom-left of Figure 3.5) with values which are inversely proportional to the density of the specific colour detected in relation to the whole image. Initially, the image is segmented into 6 main colours: red, yellow, green, cyan, blue, magenta, and black and white resulting in a total of 8 segmented regions (see top-right of Figure 3.5).
Colour segmentation uses the method of colour centroid segmentation (CSS), suggested by Zhang et al.[2]. This method transforms the 3D colour space to 2D coordinate system by using the colour triangle shown in Figure 3.6(left)(b). To create the colour triangle, a standard 2D Cartesian coordinate system is used to describe red, green and blue values and then transform it to polar coordinate system as shown in Equations 3.3.

$$\begin{aligned}
R: \quad & r(\varphi_R) = r_R, \quad (\varphi_R = 90^o, 0 \le r_R \le 255)\\
G: \quad & r(\varphi_G) = r_G, \quad (\varphi_G = 210^o, 0 \le r_G \le 255)\\
B: \quad & r(\varphi_B) = r_B, \quad (\varphi_B = 330^o, 0 \le r_B \le 255)
\end{aligned} \tag{3.3}$$

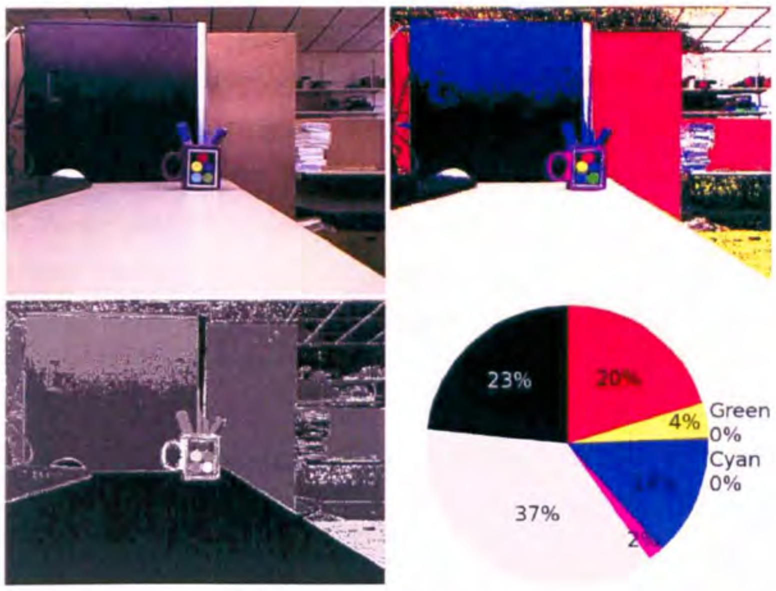


Figure 3.5: (top-left) Original Image, (top-right) Colour-segmented Image, (bottom-left) Colour density weighting, (bottom-right) Colour histogram

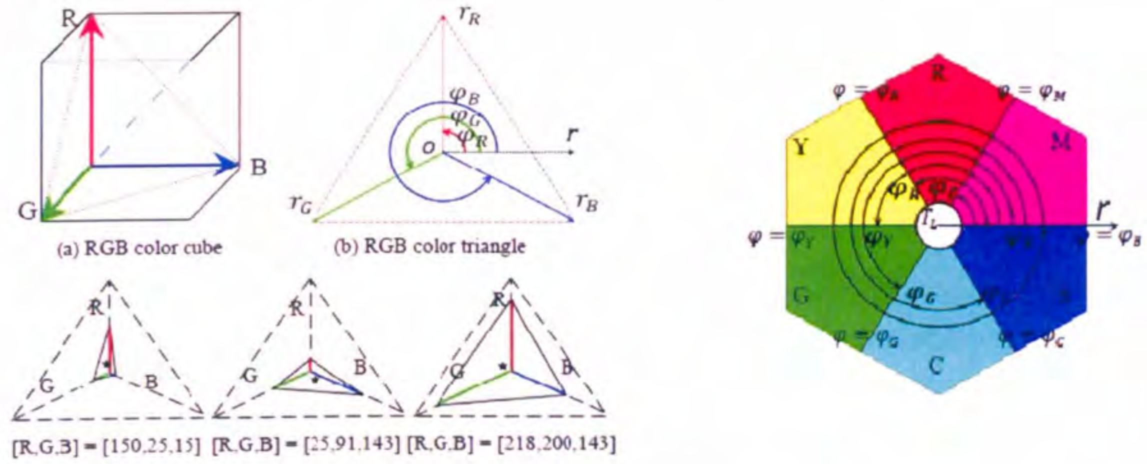


Figure 3.6: Colour segmentation - [2]

The colour triangle is created by:

1. create a standard 2D polar coordinate system
2. create three colour vectors to reflect red, green and blue colours; every vector's value range is 0, 255 with a 120^o phase.
3. generate the colour triangle by joining the three apices.

Since the R, G, B vectors direction is fixed and the values vary from 0 to 255, different colour triangles are created for different R, G, B values. This implies that each triangle will have a unique centroid (with the exception of all values being equal, with the centroid being zero). The centroids from each colour triangle are distributed along a hexagonal region as shown in Figure 3.6(right). This hexagon is divided into

7 regions: R (Red), G (Green), B (Blue), C (Cyan), M (Magenta), Y (Yellow) and L (Luminance, achromatic) regions. The colour information is directly proportional to the centroid distance from the origin. Taking (R, G, B) = (255, 0, 0) as an example would generate a centroid at the peak of the hexagon shown in Figure 3.6. When the R, G and B values are very similar to each other, independent of the value, the centroid will be very close to zero. This means that the RGB values just provide the luminance level (greyscale) with very minimal colour information. The threshold for the luminance region is defined by:

$$r(\varphi) = r_L, (0 \leq \varphi \leq 360)$$

The other six colour thresholds are defined as follows:

$$\begin{aligned}
MRegion: &\quad (\varphi_B \leq \varphi \leq \varphi_M), \quad r_M > r_L \\
RRegion: &\quad (\varphi_M \leq \varphi \leq \varphi_R), \quad r_R > r_L \\
YRegion: &\quad (\varphi_R \leq \varphi \leq \varphi_Y), \quad r_Y > r_L \\
GRegion: &\quad (\varphi_Y \leq \varphi \leq \varphi_G), \quad r_G > r_L \\
CRegion: &\quad (\varphi_G \leq \varphi \leq \varphi_C), \quad r_C > r_L \\
BRegion: &\quad (\varphi_C \leq \varphi \leq \varphi_B), \quad r_B > r_L
\end{aligned}$$

The initial thresholds for the above are set to 5, $60^o, 120^o, 180^o, 240^o, 300^o$ and 360^o respectively. These values however might need to be fine tuned to reflect better the colour segmentation. Moreover, this algorithm/filter does not distinguish between white and black as all grey scale values are considered as not having any colour information. Therefore a simple thresholding on the greyscale image is carried out and the white is reintegrated within the segmented colour image. Following this, a histogram is created to estimate the density of each colour within the image (see bottom-right of Figure 3.5). A new grayscale image is created where each colour patch is set to a value inversely proportional to the density of the colour. The more frequent the colour in the image, the lower value its weight would be (bottom-left of Figure 3.5). This is computed using Equation 3.4, where W_c is the normalized weight for the respective colour and D is the decay rate.

$$f_c(j) = (1 - W_c)^D \tag{3.4}$$

This method was found to work particularly well for tag identification which is further described in Section 3.2.1.

Colour(II) - Hue, chroma and lightness colour filters

For each 25×25 image patch, the average RGB value is calculated from all the pixels within the patch. This is then converted into a point within the HCL[2] (Hue, Chroma, Lightness) colour space [24] represented by Equation 3.5. In some colour models, such as HSV and HSL, colours that are perceptibly similar (such as those with low value in HSV model) may not be particularly close within the model. The dimensions in such model spaces map poorly to perceptual properties and encourage the use of highly saturated colours [157]. HCL is a perceptually-based colour model that mitigates these problems.

$$f_c(j) = \langle H_j, C_j, L_j \rangle \tag{3.5}$$

Orientation

Orientation can also be important in saliency detection. To produce a descriptor for the orientation within a window, the image is first convolved with a set of four Gabor filters, with orientations of $\theta = 0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}$ [24], described by Equation 3.6.

$$G_\theta(x, y) = \exp(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}) \cos(\frac{2\pi x'}{\lambda} + \psi) \tag{3.6}$$

Where:

- $x' = x\cos\theta + y\sin\theta$
- $y' = y\cos\theta + x\sin\theta$
- γ is the filter aspect ratio.
- σ is the standard deviation of the Gaussian.
- λ is the wavelength of the sinusoid.
- ψ is the phase offset.

For each orientation, the average absolute response for the filter at all the pixels within the window j is calculated as $g_\theta(j)$, described by Equation 3.7.

$$f_g(j) = \langle g_0(j), g_{\frac{\pi}{4}}(j), g_{\frac{\pi}{2}}(j), g_{\frac{3\pi}{4}}(j) \rangle \tag{3.7}$$

[2]http://www.huevaluechroma.com/081.php

Filter combination

A measure of salience is first calculated for each of the above features. The salience of a window depends not on the value of the feature for the window, but on the comparison of that value to other windows within the image [24]. The differences between two windows, w_i, w_j, for each of the features is calculated as follows:

$$\begin{aligned} d_e(i,j) &= |fe(i) - f_e(j)| \\ d_g(i,j) &= \|fg(i) - f_g(j)\| \\ d_c(i,j) &= \sqrt{\begin{array}{c}(C_i \sin H_i - C_j \sin H_j)^2 + \\ (C_i \cos H_i - C_j \cos H_j)^2 + \\ \left(\frac{L_i - L_j}{l}\right)^2\end{array}} \end{aligned} \tag{3.8}$$

The value of l in the calculation of $d_c(i,j)$ in Equation 3.8 is used to control the effect of lightness, where lightness is defined as the perceived brightness of an object compared to that of a perfect white object ranging from from dark (0%) to fully illuminated (100%)[3]. In our experiments a value of $l = 4$ is used, based on experimental evidence, reducing the influence of the lightness, relative to the hue and chroma. For each window, the similarity to each of the other windows in the image is calculated. The k most similar windows are then selected as set K. This is done, because for any window, be it salient or not, a large portion of the other windows will still be quite different from it, therefore considering only the k most similar windows should be sufficient for determining the salient regions, whilst considering all the windows would dilute the result. A value of $k = {}^{w}/_{10}$ was used, where w is the total number of windows.

It is also important to consider the distance, within the image, between two windows. If a similar window is close to the object window, then it is likely to be part of the same salient object. However, if a very similar window is physically far from the object window, this likely indicates that the object window is not part of a salient region. When calculating the salience, the difference measures $(d_c(i,j), \ldots)$ is multiplied by a sigmoid function of the Euclidean distance between the two windows as shown in Equation 3.9. x_{ij} is the Euclidean distance between windows i and j; α is the function gain and β is a shift in the x-axis.

$$\begin{aligned} s(x_{ij}) &= 1 + \frac{\alpha}{1 + e^{(\beta - x_{ij})}} \\ R_c(i,j) &= d_c(i,j) * s(x_{ij}) \end{aligned} \tag{3.9}$$

[3]http://www.huevaluechroma.com/081.php

The final saliency, for a particular scale, for window i, $S(i)$, is then calculated as presented in Equation 3.10.

$$S_c(i) = \sum_{j \in K} R_c(i,j)$$
$$S(i) = w_c S_c(i) + w_e S_e(i) + w_g S_g(i) \tag{3.10}$$

Based on experimental evidence, the weights used to control the influence of each feature were set to $w_c = 1.6$, $w_e = 1.2$ and $w_g = 1.0$.

Figure 3.7 shows the integrated saliency map approach. The bottom-up approach is carried out by applying three filters and the top-down approach is implemented by adding a face detection module to the system. It can be noted that if a face is detected, a mask(i.e. binary image) is created around the region of the detected face. This mask is then combine with the rest of the filters.

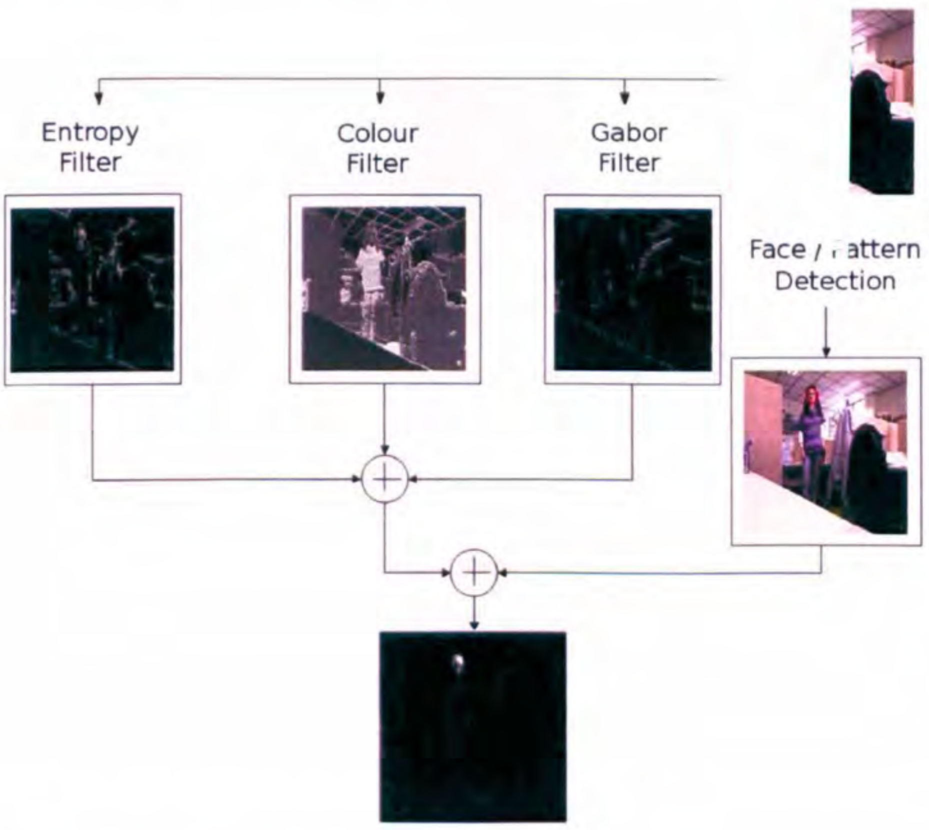


Figure 3.7: Bottom-up and top-down saliency map computation

Multiscale

This process is performed on a multiscale pyramid consisting of the full image, half and quarter scales, resizing the image each time, but retaining all other parameter values [24]. The positions of the window centres are adjusted so that they refer to the same point within the image. For each window, the final, multi-scale saliency is taken as being the mean of the saliency for the three scales. Figure 3.8 illustrates the final saliency map as a result of a sum of the saliency map at different scales.

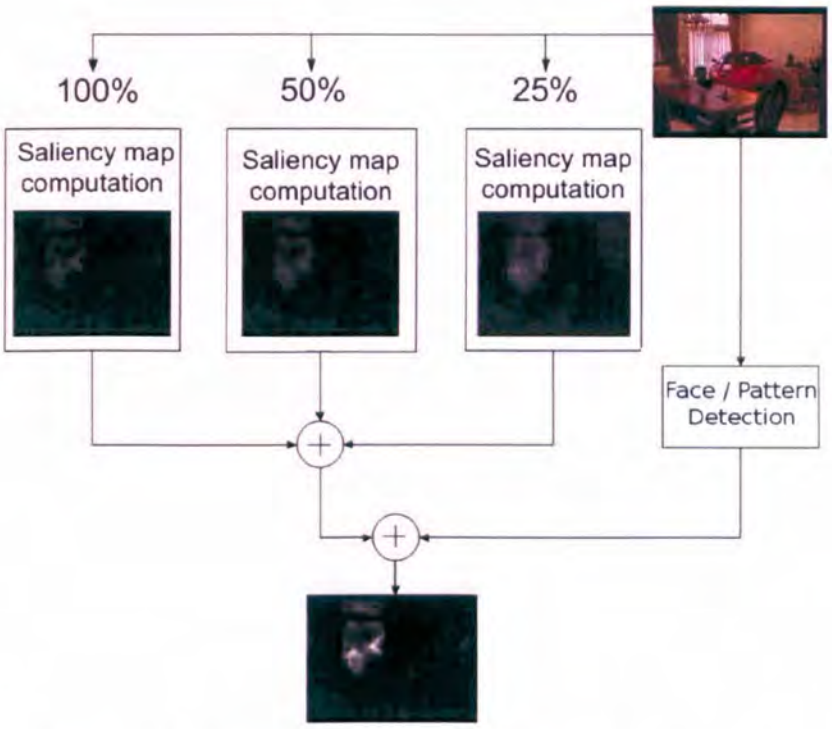


Figure 3.8: Multi-scale saliency computation

3.2 Pattern detection

Section 3.1 indicated that the system architecture proposed makes use of both bottom-up and top-down approaches for saliency detection. The bottom-up approach was presented in Section 3.1.1. The top-down approach, which was integrated in the system architecture mentioned earlier, will be presented next. In Section 3.2.1 a method developed to locate and retrieve information about objects in a home is described. This method was originally developed as a standalone device to assist persons suffering from memory and other cognitive impairments (such as dementia, Alzheimer's disease, etc.) to help them locate and retrieve information about objects in a home[4]. This method however can also be applied for object detection and localization for a mobile robot[5] operating in an ambient assisted living environment. This method uses the same principle as the face detector and can also be used within our top-down approach for saliency detection. These tags will be described next.

[4]http://www.youtube.com/watch?v=0M-_rwKvD3E
[5]http://www.youtube.com/watch?v=tAfuk-3qLME

3.2.1 Smart-home tags

This section looks at systems that employ labelling technology to identify objects and also provide our own implementation. In [158, 159] rather than adding a new tag, the existing characteristics of the object to be identified are used. Merler et al. [158] look into the challenge of recognizing food items such as a can of soup in a visually cluttered environment. They describe a vision-based object recognition system to recognize groceries in a grocery store using training image data captured under laboratory conditions. Narasimhan et al. in [159] simplify the problem by reading barcodes found on most grocery items. The system, an aid for visually impaired shoppers, is based on a smartphone. The types of labels catered for are UPC bar codes and RFID as found on items of clothing. The reading is at close range, in the range of millimetres, and the user must be guided towards the items. The two systems described above rely on reading the label and retrieving the information provided by the manufacturers of the labeled product. The systems described next do not use the manufacturer label but rather a proprietary labeling and information content system. There are a number of RFID-based localization and navigation systems found in the literature. Some are considered here. Hallberg et al. [160] localize household object which have been RFID tagged to assist persons suffering from dementia; the localization algorithms employ Received Signal Strength Intensity (RSSI) on the reader. Kulyukin et al. [161, 162] also employ RFID tags for an indoor assisted navigation system; the RFID tags are dispersed in the environment and the reader located on a mobile object identification and retrieval in a smart home robot platform. Moving away from RFID technology, localization systems that use a 2D visual pattern are considered. Labelling technology and label reading technology has evolved from the simple one-dimensional bar code read (scanned) by an optical decoder to two dimensional matrices known as a data matrix. The technology employs a machine-readable data code that is printed onto a tag which is attached to an object. There are a number of proprietary and open-source systems available; differing in the representation and encoding of the data which in turn determines the method used to read the code. The Microsoft Tag is mentioned here as it uses colour to increase data capacity. The High Capacity Colour Barcode (HCCB) uses clusters of coloured triangles. The data density is increased by using a palette of 4 or 8 colours for the triangles. In a Microsoft Tag, the HCCB contains a hyperlink, which when read, sends the HCCB data to a Microsoft server, which in turn returns the publishers intended URL. The Tag reader directs the users mobile browser to the appropriate website. An assistive device employing machine vision, was described by Coughlan et al. [163, 164]. The focus is navigation for the visually impaired; employing a mobile phone camera to detect the colour targets situated in the environment. The use of coloured targets with specific patterns allows the target to be detected at a distance and in seconds. Another common tag which is becoming very popular is the QR Code. The HCCB and QR code are used as shortcuts to mobile webpages. Figure 3.9 and Figure 3.10 show the HCCB and QR Code respectively, both pointing to the author's PhD blog website[6].

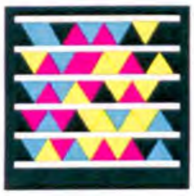


Figure 3.9: Microsoft Tag - HCCB

Figure 3.10: QR-Code

Although these tags are becoming quite popular to use on smart phones, at the time of testing it was found out that these tags were not particularly suitable for an ambient assisted living application. Due to the relative complexity of the image and therefore the software used to decipher them, other alternatives had to be found. It was noted that for such tags to be useful, the captured image would need to have the tag in the centre of the captured image. Also, the tag need to take most of the image area. Moreover, the camera needs to be relatively stable. Due to this tags to meet the specific application needs had to be designed as will be discussed next.

The proposed tag relies on computer vision techniques to locate a tag on an object when it is within the field of view of the device. The Viola-Jones object detector (described in Section A.4.1) is used in order to detect tags. Colour segmentation techniques (described in Section 3.1.1) were then applied to identify the detected tags. The tag is conceptually similar to 2-dimensional labels that hold item or product information such as the QR-code or the Microsoft tag mentioned above. However there are two main differences: the detection range and robustness to tremor/jitter. The tag is a 2D colour printed pattern with a detection range and a field of view such that the user may point from a distance of well over one metre with a camera resolution as low as 320×240, with distance increasing with higher camera resolution. The target users are elderly persons, particularly those with memory and/or physical impairment. Robustness will ensure tremor or effects of an unsteady hand are removed. To satisfy these requirements, the range and the field of view (FOV) are such that the user (or robot) may point from a distance of over one metre. The tag will be detected as long as the FOV is within $\pm 30^o$ and audio feedback will guide the user or robot towards the object until the user or robot faces the object. A small foot print for the tag, $57mm \times 70mm$, will ensure it can be placed on smaller household items. The detection range is constrained by the camera resolution and the tag size. The range and FOV will depend on the camera characteristics, but can be increased by using

[6]http://www.raphaelgrechphd.blogspot.co.uk/

a higher resolution camera, or increasing the tag size. Information about the object pointed to is contained within a database held on the hand-held device. It is envisaged that a dedicated standalone device is developed, however the systems software may operate on a generic PDA or mobile phone with a camera.

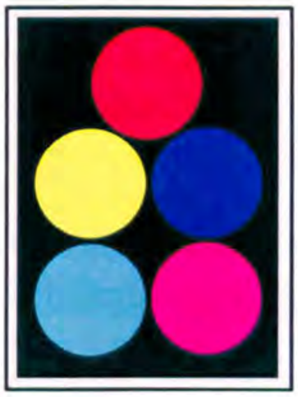


Figure 3.11: Proposed Prototype Tag

The prototype tag is shown in Figure 3.11. It comprises of five circles (dots) where each circle can take on any of the 6 colours - red, green, blue, magenta, cyan, and yellow. The border, surrounding the circles, allows the tag to be detected, akin to the finder pattern in a data matrix and facilitates localizing. Orientation is determined with a marker indicating a point of reference. In this scenario, the colour order is important and repetitions are allowed. This provides for n^r permutations, where n is the number of used colours and r the number of used dots. In this case, there are $6^5 = 7776$ permutations. If repetitions are not allowed then the number of permutations is cut down to 720 according to

$$\frac{n!}{(n-r)!} \tag{3.11}$$

In the context of object recognition for ambient assisted living, this provides a sufficient number of unique patterns to identify all required objects. The rationale for the tags design, from a computer vision perspective, is ease of production, using any colour laser printer; the use of primary colours facilitates segmentation, and orientation. The requirements are for the tag to be detectable from at least one metre with a 320×240 pixels camera and up to two metres with a 640×480 pixels camera (the target is detected up to $^1/_{100}{}^{\text{th}}$ of image size). The detection of the tag pattern in an upright position allows for a tilt/skew of up to $\pm 30^o$. Detection must be robust under different lighting conditions and in a visually cluttered environment. The data contained within the tag is extracted with a camera mounted on a small hand held device such as a smart phone so that standard computer vision techniques can be employed. Tag data extraction is performed in two steps: the detection of the overall tag shape followed by a segmentation and extraction of the coloured circles. In the case of the robot, the tag is detected using a camera mounted on the robot. If detection is done using Kinect or similar technologies, the distance of the tag from the robot can also be measured.

Retrieving data from the tag necessitates three steps, namely tag detection, tag identification and meta-data extraction. For this implementation, a method which is fairly robust and very fast to run was needed. For tag detection the same methodology as that for the face detection discussed in Section 2.2.1 was used, this time however training the Haar boosted classifier on tags rather than faces. Training is relatively time consuming, however, considering that training is only performed once during setting up, this time constraint is acceptable. Once the tag shape is detected and recognized successfully, tag identification is obtained by carrying out colour segmentation as described in Section 3.1.1. This would then prompt a database query in which database information pertaining to the object is kept. In order to train the classifier successfully, numerous well-segmented data were required. This would generally mean thousands of object examples and tens of thousands of non-object examples (in our case, tag and no tag images). Given the rigidity of the proposed tag however, much less images for training were required and which still provided impressively good results. Also, in order to have good classifiers, data should not be mixed. If both tilted and upright tags need to be identified then the training data should be divided and two classifiers are created[46]. For proof of concept case, only upright tags were considered and it was assumed that these tags will be always affixed in an upright position. For good training data should also be well-segmented and consistently boxed. Sloppiness in box boundaries of the training data will often lead the classifier to correct for fictitious variability in the data [46]. Our training set consists of 212 positive images (some containing multiple tags) and 384 negative images. Positive images refer to images in which tags are present and negative images are images which do not contain tags. The tags were then located and cropped from the positive images. Moreover, only six of the possible 7776 tags were used for training. The rationale for this choice was to investigate the performance with a low training set. Two training sets were created using different cameras for cross validation. The first dataset, set *I*, comprised images generated using two different hand held cameras and a robot camera. The second dataset, set *II*, was generated using the robot camera. During evaluation, validation was performed with dataset *II* when training was carried out with dataset I and vice versa. While the first dataset (set *I*) consisted of images of tags located randomly in the environment; dataset *II* was generated with the robot camera located at the node positions in Figure 3.12. The test image was captured at 30^o interval.

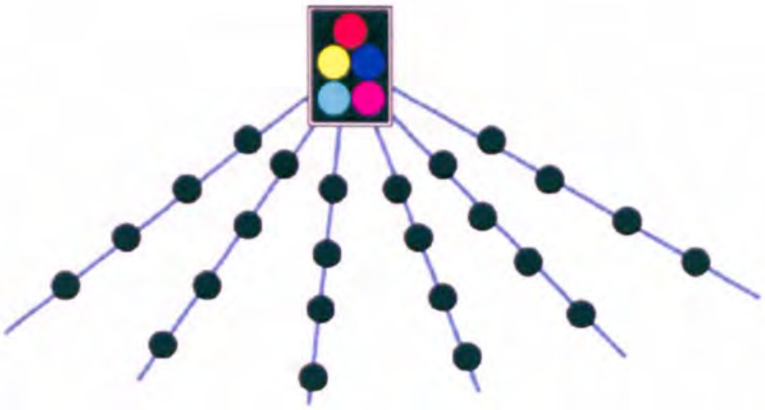


Figure 3.12: Test pattern generation

Experimental setup and results

Software development was carried out using the OpenCV library [46] on a laptop connected to a robot and to a USB web camera. OpenCV is an open source development library written in C, and it is specifically suited for real time applications. Tests were carried out to investigate the praticality of our system. These tests involved: a) varying the distance and the orientation to determine the maximum detection range of the system, and (b) employing different cameras and light conditions to assess robustness and (c) exposing the system to severe jitter to reflect the condition of the actual user for whom this application is intended. Additional tests (d) were carried out to investigate performance with the tag applied to non-flat surfaces.

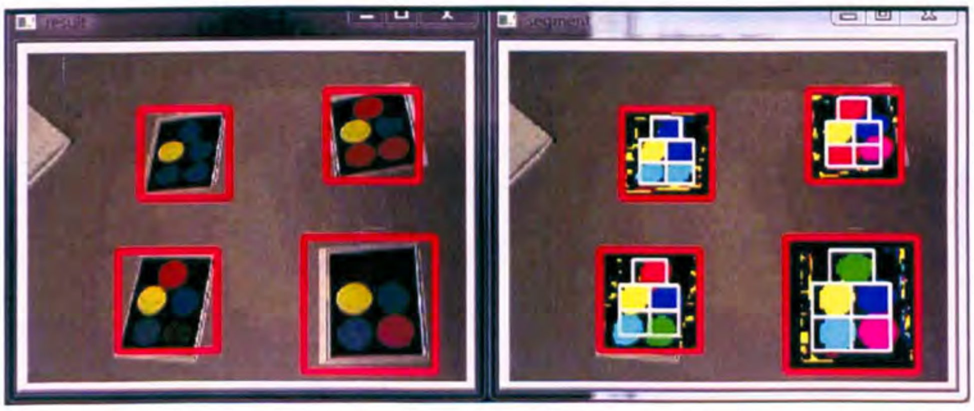


Figure 3.13: Tag detection (left) and identification (right)

In the experimental setup, the tag size was $57mm \times 70mm$ and the camera was set up to a resolution of 320×240 pixels. In practice, the system is able to detect tags as low as 10 pixels $\times$ 10 pixels; the limitation is to allow for colour segmentation/detection within the small white boxes. Figure 3.13 shows results with four tags. In Figure 3.13(left), the blue and cyan are adjacent in the colour space however these were successfully identified, as shown in Figure 3.13(right). Tag detection capability of segments in the images affected by skew and affinity can be seen in both Figure 3.13. In the prototype, the detection area is fixed and contained within the small white boxes thus restricting the angle of detection to ±30. This can be improved by employing a rotation detection algorithm. The top right tag and bottom left show the current skew limitation; if the skew angle were to be increased further these tags will not be detected.

Detection range and orientation

During the testing phase it was expected that all the six patterns on which the training was carried out would be easily recognizable. A further six images comprised the independent test set. The independent test data were successfully recognized. Figure 3.14 shows the detection percentage at a given distance and orientation respectively.

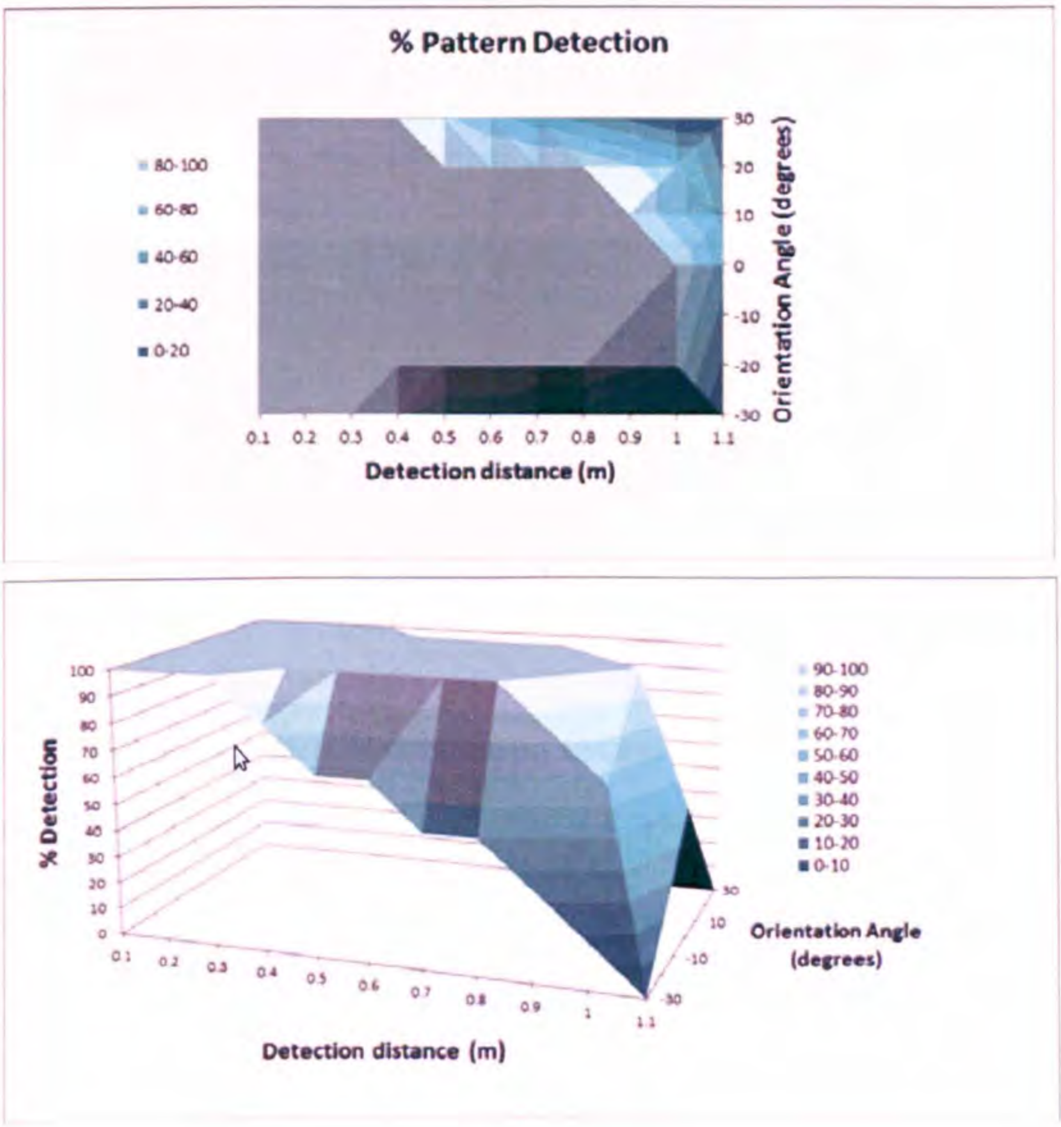


Figure 3.14: Detection percentage at a given distance and orientation

The tests were carried out using the robot placed at a fixed distance and angle from the tag to be detected. Increasing detection range can be achieved with a higher resolution camera. It is also worth noting that for a 640 × 480 pixels resolution detection was achieved in real time. However the 320 × 240 pixels resolution, was used to enable the system to be integrated into off-the-shelf embedded systems with lower camera resolution. The light intensity was varied from good lighting to dim lighting and the pattern was again successfully recognized and colour segmented as shown in Figures 3.15 and 3.16.

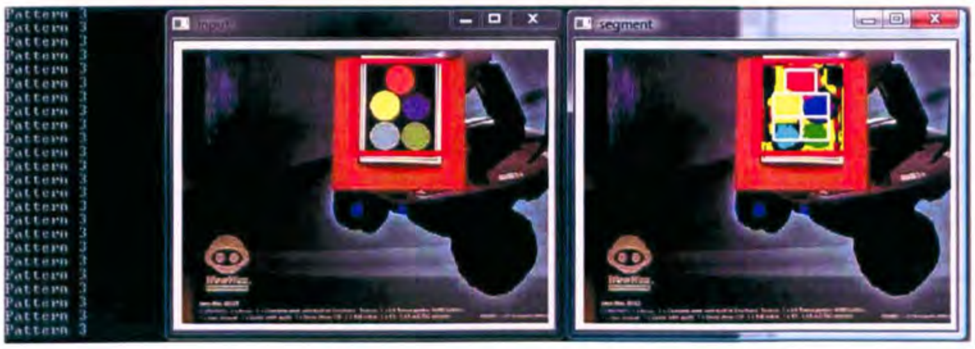

```
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
```

Figure 3.15: High light intensity

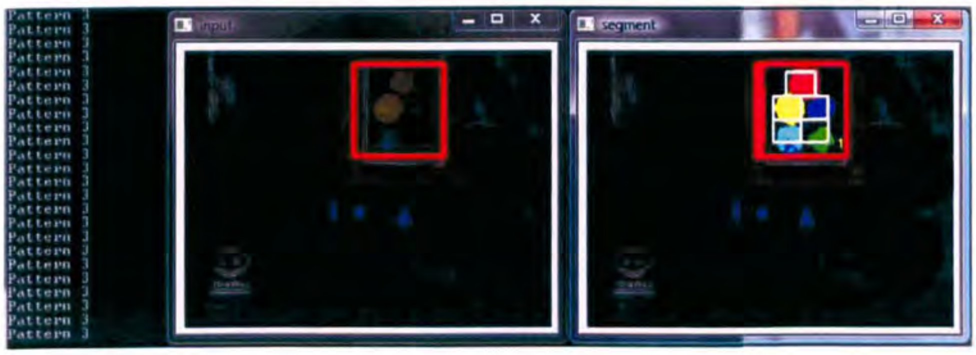

```
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
Pattern 3
```

Figure 3.16: Low light intensity

Detection under conditions simulating tremor (jitter)

As an assistive device, detection performance must be maintained when subject to tremors: as when used by a person suffering from Parkinsons disease or any other condition in which the user is unable to hold the camera steady. A 3D accelerometer was attached to a webcam to measure and coarsely classify the simulated tremor. The experiments were repeated with jitter included to simulate an unsteady hand. The results are shown in Figure 3.17: the left image clearly shows the blurring and the one on the right is the real time detection of the patterns. The tags were successfully detected with various degrees of jitter. The detection limitation comes from blurring caused when velocities are high. It can be seen that system performs well under severe jitter.

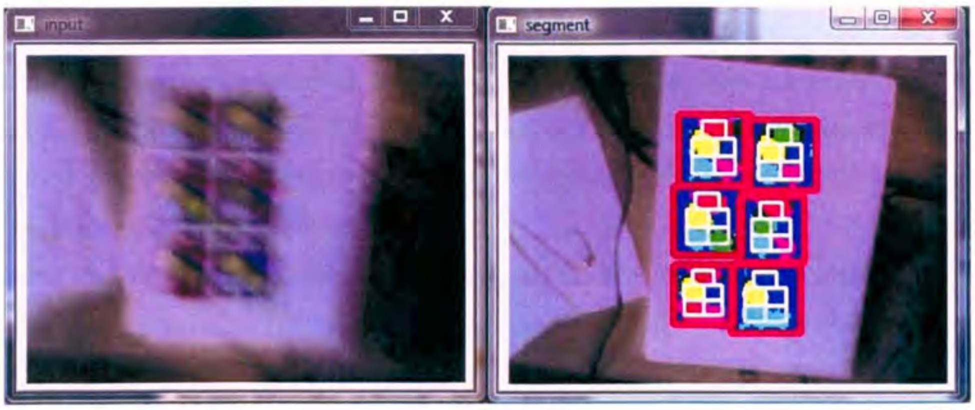


Figure 3.17: Detection under conditions of jitter, blurred image (left) and detected tags Detection on Curved Surface (right)

In a home environment, the tag may be fixed to a number of objects; some objects might have a curved shape. When the pattern is fixed to a curved surface the circles will become ellipses, however, the Haar classifier, will detect these patters, even though training was carried out on flat surfaces. The result can be seen in Figure 3.18.

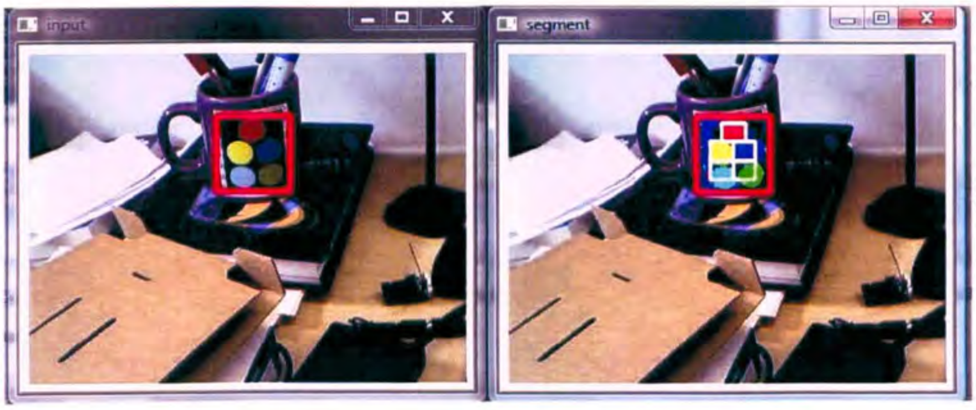


Figure 3.18: Tag on a curved object

3.3 Summary

This chapter discussed a method which uses a mix of top-down and bottom-up approaches, to make robots aware of their dynamic environment and capable of identifying both known objects and yet unknown salient objects. Such robots will generally work in cluttered, hard to train for, environments. By applying computer vision and machine learning techniques, robots are endowed with the capability to perceive and be made aware of salient regions within their environment. Some of the proposed methods are inspired by how humans look at interesting areas within their environment. This chapter also looked into the design of identification tags which allows robots working in ambient assisted living to guide people to specific areas and possibly also providing them with metadata about the object to which tag is attached. Results show that tagged objects can be detected at over one metre using image resolution of 320×240 pixels and tags of size $57mm \times 70mm$ even in poor lighting conditions as well as under conditions of jitter. A combination of larger tags and higher resolution images would increase the detection distance.

The next chapter describes how the robot capability is further enhanced by applying an unsupervised machine learning mechanism allowing robots to extract previously undetected salient information from an incoming robot video stream. This then feeds into a visual memory which in turn can be used to provide robots with a more selective searching capability.

"The difference between false memories and true ones is the same as for jewels: it is always the false ones that look the most real, the most brilliant."

Salvador Dalí, *Surrealist Spanish painter (1904 - 1989)*

Chapter highlights:
This chapter presents an efficient method for the real-time and online evolution of very concise and compact robot visual memories stored in a flexible graphical representation using a Growing Neural Gas (GNG) network.

4 Visual Memories

Chapter 3 looked at how visual perception can be implemented on robots, making them aware of their environment by using a combination of top-down and bottom-up approaches. Whilst some amount of pre-programming or training can be applied to robots to provide them with the ability to detect certain objects, it is not possible to predict all the situations that the robots will have to deal with in complex conditions [165]. In this chapter the robot's performance over the methods suggested in Chapter 3 are extended by focusing on methods to endow the robots with learning capabilities.

An efficient method for learning and memorising online and in real-time an environment from a sequential input video stream in a very concise and compact manner using robots is proposed. The idea is to generate a graph of small size as a reduced representation of the environment that could be easily shared among robots or a distributed set of computational nodes and easy to grow in cooperation. In this context, real-time means that the sampling input frequency is sufficiently high to reflect any changes within the environment scene. The encoding and storing of the visual appearance of locations by robots memorising scenes aims at creating a general understanding of the environment which is easily understood and referenced by humans. Robots have their own memory represented by a graph with nodes encoding the visual information of a video stream as a limited set of representative images. This approach allows an environment to be topologically labelled, for mapping and self-localization purposes.

As discussed in Section 2.1.2, biological systems have proved to work very efficiently, mostly attributed to millions of years of evolution. Therefore, it is thought that similarly to what happens with humans [65], only salient images, called keyframes, are stored, thus requiring less processing power and storage space; however still all the relevant information is maintained. Methods are studied and developed to allow robots to learn keyframes describing their environments. These keyframes are dynamic and evolve so as to replicate what is *currently* unfolding in the environment. A robot needs to be able to learn and adapt these keyframes which could be shared between other robots and other humans, such as security guards, using such system.

Section 4.1 looks at the human memory, followed by methods to replicate it on robots by using unsupervised learning techniques, discussed in Section 4.2. As will be described in the coming sections, the robot's visual memories are incrementally built using the growing neural gas (GNG) self-organising model. GNG was chosen because it provides flexibility and portability, by dynamically building a representative graph of the input space, in this case a video or scene video sequence. GNG networks are self-organising neural networks that can dynamically adapt their reference vectors and topology. Frames are sequentially processed by the GNG, automatically generating nodes, establishing connections between them and creating clusters dynamically. GNG forms clusters of similar images in video streams, averages of which are used as keyframes representing the (potentially) distinguishing parts of the environments shown in the streams. This is then all combined in Section 4.3 so as to have visual memories coming from robot video streams.

4.1 Human visual memory

Visual memory is the ability to recall information that was previously presented in visual form such as images, maps and lists of words [166]. It is therefore that part of memory preserving some characteristics of the senses pertaining to visual experience. Research, carried out in biology and psychology, studies how humans and animals rely on visual memory to interact between themselves and their environment. Human memory can be divided into short-term memory (STM) and long-term memory (LTM) [166]. STM is used to temporarily retain information in an active, readily available state for a short period of time, whereas LTM holds information for much longer and in some cases, even permanently within the brain. Humans seem to rely on years of prior experience in making decisions based on what their eyes see. People can generate robust visual memory representations of natural scenes, especially from features of objects and their locations. The encoding of stable visual information relies on attention, search strategy, and the ability to retrieve encoded memories. The relationship between visual perception, attention, and memory, all dynamic and serial cognitive processes, extends over space and time. Since visual encoding is suppressed during saccadic eye movements [167], visual inputs from the scanning of a scene are partitioned into discrete episodes, weighted by their saliency and held together in a memory buffer. Visual memory supports the accumulation of information from scores of individual objects in scenes via iconic, short-term and long-term storage strategies. Iconic memory is precise, but volatile, brief ($< 300ms$) and highly susceptible to interference from new sensory processing. If the scene is removed or perceptual processing otherwise interrupted, sensory persistence decays quickly and is not integrated from one view of the scene to the next. Visual short-term memory (VSTM) has a limited capacity of 3-4 objects and less spatial precision than iconic memory although it can be maintained for seconds and across saccades. Visual long-term memory (VLTM) allows accumulation of visual information from scores of individual objects, leading to the retention of the scene gist rather than to the retention of visual details in photographs over relatively long periods of time. People can generate robust visual memory representations of natural scenes, especially from features of objects and their locations. The encoding of stable visual information relies on attention, search strategy, and the ability to retrieve encoded memories. Search strategies help keeping track of targets and efficiently guide attention to the ones that appear in predictable locations within real-world environments [168]. Directing attention to an object/scene allows the formation of a coherent visual representation and the consolidation of that representation into a more stable VSTM. Recent studies by Konkle and colleagues [169], have shown that observers can store thousands of objects images as well as scenes in VLTM with high fidelity, suggesting that scenes and objects may be best treated as entities at a similar level of conceptual abstraction, providing the semantic structure necessary to support recognition and memory of visual details. Data in [169] also indicated that observers' capacity to retrieve visual information encoded in VLTM depended more on conceptual structure than perceptual distinctiveness [170]. Memory for the spatial layout of a scene and memory for specific object positions can efficiently guide search within scenes, as it helps to keep track of objects that have already been examined and efficiently guides attention to targets that appear in predictable locations within real-world environments [168]. As suggested in [171] there are two alternative theories of how memories are encoded in the brain. One theory suggests that a single memory is stored in a distributed fashion with bits and pieces distributed across millions of neurons. The alternative view, which has gained more scientific credibility, holds fewer neurons, numbering in the thousands if not less and constitute a sparse representation of the image. These small groupings of cells may represent many instances of one thing. This idea is generally known within the research community as "grandmother cells". The aim of this research is to replicate the latter theory into an artificial robot memory.

4.2 Robot visual memory

As already mentioned in Section 2.4, various methods to store visual experience exist within the robotics domain especially for localisation and navigation purposes. This thesis mainly focused on the use of GNG self-organising model for visual memory and reasons for the use of such model will be given in the coming sections. Various input sensors and other possible methodologies were also suggested in the literature. Some of these methods are further described in Appendix Section A.5.

GNG was also considered by other researchers [172, 66, 165, 173] and even though these studies offer some relevance to the problem being tackled in this thesis, none seem take into account the HRI aspect of the application. Moreover, as is the case in [165], the main concern to address with using a database is the issue of memory management and finding efficient ways of limiting the memory size as the image database grows large.

Cao and Suganthan [172] proposed the use of GNG networks to integrate multiple frame difference features to efficiently detect camera shot boundaries (scene changes) captured in the video with special emphasis on temporal segmentation. The segmentation method involves computing multiple difference measures for each frame pair in a sliding window, based on a combination of histogram-based comparison and motion compensated block-based comparison, and then utilises the unsupervised GNG. Kirstein et al. [66] propose an appearance based object recognition method, also considering short term and long term memories. Kit et al.'s [173] method uses GNG together with a low dimensional representation of visual features to rapidly identify regions of change within an image. Saponaro and Bernardino [165] studied a way for personal robotic assistants to deal with dynamic and uncertain information when they are deployed in private and public settings. They try to address the problem of creating visual memories of salient objects arising in a certain environment and the ability of recalling them at a later stage. In their method, images are compared using a histogram intersection technique by using the hue and saturation components in the HSV/HSI colour space. They start with the assumption that the robot is initially unaware of its environment and start building a database consisting of visual memory classes.

As already stated in Section 1.1, the aim of this thesis is to represent the environment in which the robot works in an efficient and compact way, with the memorised data being easily understood by people. The methodology to achieve this is described next.

4.2.1 Learning strategy

Section 4.1 described that human memory does not retain all details of a scene and memorised images can be blurred or somewhat unclear. It is rather impossible for someone to learn and remember all the details, however, sufficient information is retained to recall relevant information from memory about a part of a scene [174]. In the case that something of importance is noted and further data gathering is required in that area, supervised learning or reinforcement learning could be used by guiding the robot to the area of interest and providing it with feedback, similar to what is done with children. Children are attracted by objects which are more interesting than others and which they would instinctively learn [175]. Besides that, parents and educators channel them and guide them to learn things which are also important for their survival however which require schooling (education and training), such as reading or playing an instrument [176].

This study focuses mainly on unsupervised learning methods for robots to learn their environments, synonymous to the instinctive learning mentioned above. Unsupervised learning is performed using only data without any teaching signals [3]. The main characteristic of learning-based approaches is their ability to adjust their internal structure according to input and respective desired output data pairs in order to approximate the relations implicit in the provided (training) data, thus elegantly simulating a reasoning process. Methods for clustering the newly learnt information are studied. Self-organising map (SOM), neural gas (NG), growing cell structures (GCS) and growing neural gas (GNG) are well known unsupervised learning methods [3]. SOM is one of the most successful and widely used clustering network suggested by Kohonen [177] using a set of heuristic procedures which is not based on the minimization of any known objective function. SOM performs clustering while preserving topology where the number of nodes and the topological structure of SOM is designed beforehand [3]. When working with SOM, one has to keep in mind that SOM has a forced termination, unguaranteed convergence, the procedure is not optimised and the output is often dependent on the sequence of input data [178]. Van Hulle [179] examines SOM in detail together with other networks such as the NG, which is also tackled in [178]. Du [178] states that unlike the SOM, which uses predefined static neighbourhood relations, the NG determines a dynamical neighbourhood relation as learning proceeds. The NG is an efficient and reliable clustering algorithm, which is not sensitive to the neuron initialisation. In NG, the number of nodes is fixed beforehand, but the topological structure is updated according to the distribution of sample data [3]. Furukawa [180] proposed an extension to the SOM, calling it SOM of SOMs. He also indicates that this method can easily be generalised to any combination of SOM families including cases of NG. Moreover GNG, originally introduced by Fritzke [181] is a NG which grows over time. The growing mechanism of GNG's hidden layer is similar to the one of GCS, the main difference being that GNG uses a type of competitive Hebbian learning (CHL) to modify the topology structure of the graph. This allows GNG networks to form neighbourhood graphs embedded in a space, whose dimensionality varies by adapting to local, data distribution profiles. Apart form the aforementioned differences, GNG and GCS are structurally the same [182]. Whereas GCS does not delete nodes and edges, GNG can delete nodes and edges based on the concept of ages. Furthermore, GCS must consist of k-dimensional simplices whereby k is a positive integer chosen in advance [3].

GNG was proved to be superior to existing unsupervised methods, such as self-organising Kohonen maps (SOM), K-means and growing cell structures [183, 184, 185]. In a GNG graph, nodes can be disconnected whilst the network is evolving, creating a separation between uncorrelated nodes. The number of nodes need not be fixed a priori, since they are incrementally added during execution. Insertion of new nodes ceases when a set of user defined performance criteria is met, or alternatively the maximum network size is reached. The algorithm iteratively learns to identify similarities of input data and classifies them into clusters. Florez et al. [183] conclude that networks having an evolving topology adapt better to the input manifold than any network with pre-established topologies. More recent work [3] also corroborates the conclusions mentioned in [183, 182]. Sasaki et al. [3] study the use of GNG, comparing it with other unsupervised methods to be used for intelligent robot vision using a range imaging camera.

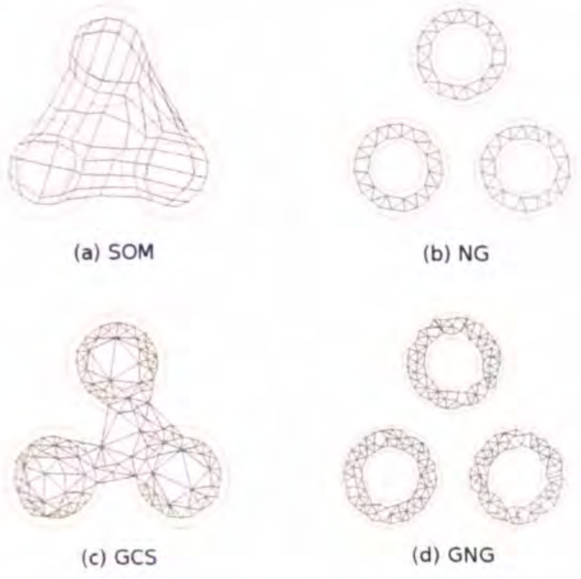


Figure 4.1: Comparison among (a)SOM, (b)NG, (c)GCS and (d)GNG [3]

Sasaki et al. [3] conducted comparison of these methods using sample data of three rings. Figure 4.1 shows the final stages of SOM, NG, GCS and GNG after 50000 iterations, respectively. Table 4.1 shows the comparison of evaluation values among SOM, NG, GCS and GNG. The computational time of GCS is the shortest because the original GCS does not delete nodes and edges. NG needs much more computational time than others because the algorithm of sorting of nodes is adopted every iteration.

	Calculation Cost (ms)	Nodes	Edges	Deleted Edges
SOM	2100	100	180	0
NG	8600	100	193	280
GCS	1200	169	501	0
GNG	1900	168	369	392

Table 4.1: Comparison of evaluation values [3]

Following the results provided above, and considering the need for an unsupervised learning technique to cater for the application, the results and conclusion presented in [182] indicate that GNG and GCS provide very satisfactory results. Considering the results provided in [3] and also taking into account separate clustering, SOM and GCS are discarded because they do not provide separate clustering. NG has a fixed number of nodes and is very slow compared to its counterparts. On the other hand, GNG provides a good performance in clustering capability as well as computational cost making it the unsupervised learning technique of choice.

The GNG algorithm

GNG was originally introduced by Fritzke [181], as an unsupervised learning technique where no prior training is needed. The system starts with two linked nodes; new nodes are inserted at every fixed number of input cycles up until the maximum number of allowed nodes is reached. Connections between nodes are also inserted and removed adapting the network topology. Moreover, nodes which are disconnected are removed thus allowing for new nodes to be inserted in a better position within the topological map. This results in a network having a topological structure composed of N nodes in K clusters connected by edges closely reflecting the topology of the feature distribution. The GNG algorithm operates as shown in Algorithm 1 with further detail provided in Appendix C. The GNG network is specified as:

- A set N of nodes (neurons). Each node $n \in N$ has its associated reference vector $\mathbf{w}_k$ belonging to the input space (80×60 greyscale images).
- A set of edges (connections) between pairs of nodes. These connections are not weighted and its purpose is to define the topological structure. An edge ageing scheme is used to remove connections that are invalid due to the adaptation of the node during the learning process.

The conventional way of training GNG entails having a training dataset from which items are randomly selected and fed into the network. The GNG algorithm was used in

[183, 186, 187] to map 2D nodes onto an image. The generated map was then employed for visual object recognition and categorization. GNG is ideal due to its incremental learning characteristic, the insertion of new nodes where required and the capability of representing newly occurring data. This generally ensures that the GNG evolves in a distributed manner and is more likely to represent the input data more accurately. GNG suffers from the conventional initialisation problems, i.e. every time the GNG algorithm is run it might evolve slightly differently, depending on the initial seeding and also on the way the node weights are adjusted.

Algorithm 1 GNG Algorithm

```
Set two nodes containing random values, age edge = 0, error = 0

while (Stopping Criterion = false) do
  - capture an input image vector x
  - from all nodes, find winning node $s_1$ and second best node $s_2$
  - increase the age of all the edges from $s_1$ to its topological neighbours
  - update the error of $s_1$
  - move $s_1$ and its neighbours towards x

  if ($s_1$ and $s_2$ are connected by an edge) then
    - set the age of the edge to 0.
  else
    - create an edge between them.
  end if

  if edges are older than age threshold then
    - remove edges
  end if
  - remove isolated neurons

  if (current iteration is a multiple of $\lambda$) and (maximum node count = false) then
    - find node $u$ with largest error.
    for all neighbours of $u$ do
      find node $v$ with largest error
    end for
    - insert a new node $r$ between $u$ and $v$
    - create edges between $u$ and $r$, and $v$ and $r$
    - remove edge between $u$ and $v$
    - decrease the error variables of $u$ and $v$
    - set the error of node $r$
  end if
  - decrease error value of all nodes
end while
```

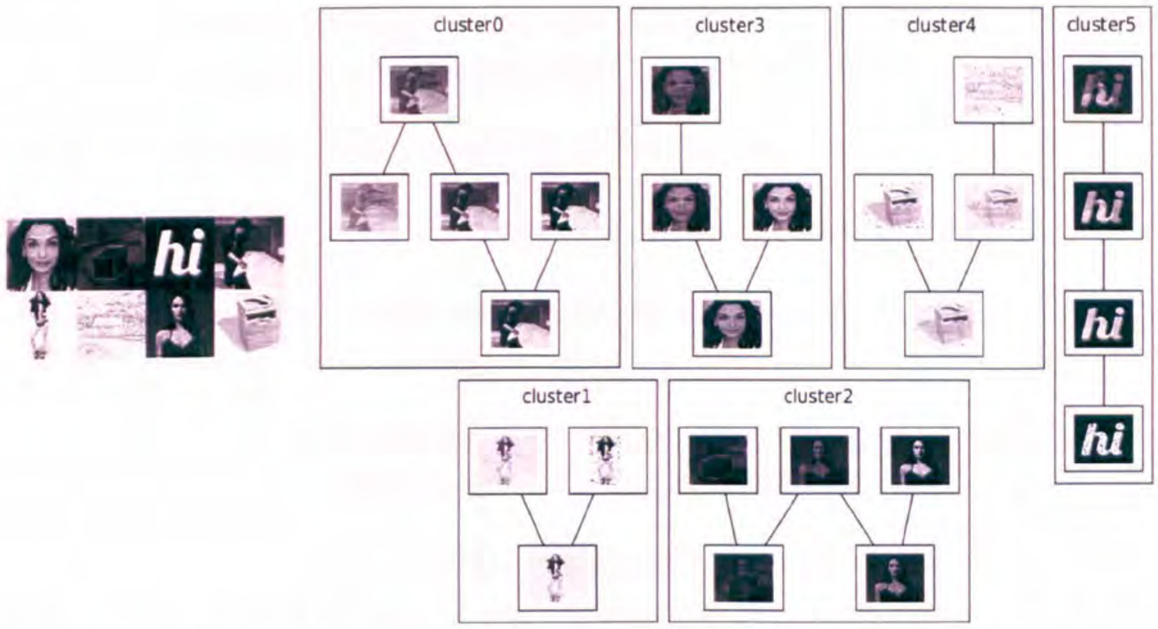


Figure 4.2: Training Images (Left) and Generated GNG Clustered Nodes (Right)

Figure 4.2 illustrates an example of how GNG structures the network topology. As will be further discussed in Section 4.3, rather than mapping 2D nodes onto an image, each node consists of an 80×60 pixel greyscale grid representing an evolving memory image. In this example, a fixed set of training images and a maximum number of nodes have been used. After a random iteration of the trained images illustrated on the left of Figure 4.2, GNG creates the links between the nodes and the clusters, which are shown on the right hand side of Figure 4.2. One can note that some of the images were merged together. This is mainly due to the feature vector used. In this case the feature vector is the greyscale pixel value, so images having similar grayscale distributions tend to cluster together. The more defined the input feature vector the better the classification is expected to be. Details on feature vector selection is discussed next in Section 4.2.2.

4.2.2 Feature vector selection

Once the most appropriate learning network is chosen, the next step is to decide on the best feature vectors to use for the training phase. Given the application described in Section 1.1, the generated robot memory has to be easily understood by humans. Moreover, a user would not necessarily be a technical person in the field of robotics or related areas. Therefore, the outcome should be as understandable and 'natural' as possible to the layperson. Although not restrictive, the following assumptions can be made:

- The robot camera motion is constrained (e.g. image always upright relative to the ground).

- robot visits same places in a similar fashion (e.g. robot moves in centre of corridor most of the times / unless restricted), and
- persons using the system are able to identify objects even in blurred images.

With these given assumptions, the following research questions are posed:

- Is there a relatively simple way to perform image correspondence which would still provide an understandable result?
- To what extent is complex feature extraction necessary to reach both real-time and good enough performance for HRI applications?

The most common image descriptors generally include colour, texture, shape, motion and location amongst others [188]. As shown in Figure 4.2 and considering the above assumptions and requirements, it might be argued that sufficient information can be stored in memory using a reduced size 80×60 pixel greyscale image. A counter argument could be that other, more descriptive features, could be used to train the GNG rather than the scaled greyscale images. Another possible option could be that of feeding in a higher semantic level to the GNG. This, however, requires an object detection stage prior to feeding the information to the learning network. This could work well in an environment where contents are known and somewhat expected, however, it would completely fail in a totally new and unseen environment. Feature extraction (information provided in Appendix Section A.3.4) is still considered as a relatively time consuming process. As an example, the best optimised implementation of SURF based recognition technique found in literature [189] claims a time reduction from 39 seconds to 780 milliseconds and is aimed to run on small embedded robot platforms with limited processing resources.

Keeping the application in mind, unless the selected features are highly descriptive, and since only these features will be memorised, there is no way to revert back to the original image. This would make it impossible for a human to understand straight away what the robot has memorised. Due to this, interest points such as corners or other similar specific descriptors were not considered as feasible options. A reduced size 80×60 pixel image was therefore used. Experimental analysis to strengthen the argument are provided next.

4.2.3 Experimental analysis

Looking at Figure 4.3, a simple visual comparison shows similarities between the generated nodes (memory images) and the frames of the original input images. This gives humans sufficient information to recall a scene within the environment. However, for

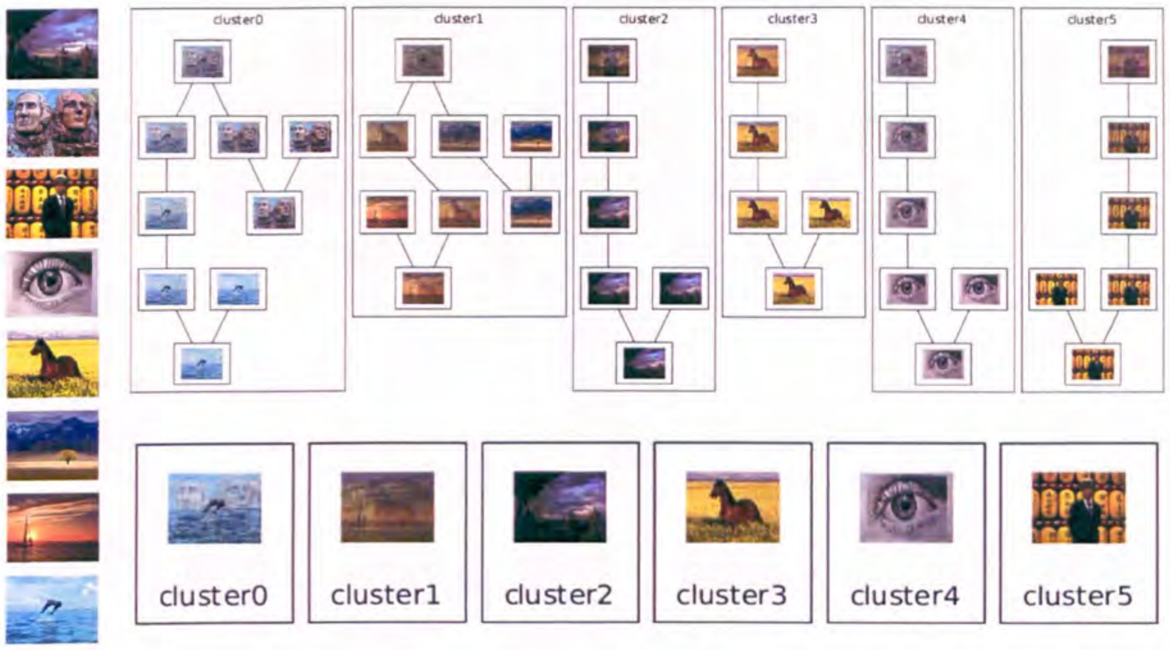


Figure 4.3: Training Images (Left), Generated GNG Clustered Nodes (Top-Right) and Keyframes (Bottom-Right)

a robot, a quantitative assessment is required. A metric to measure the similarity between what is currently seen and what was memorised earlier is required. In Figure 4.3 each node contains RGB colour values. One can note that similar images are clustered together, based on the similarity measurement criteria. Figure 4.3 (Bottom-Right) shows the generated keyframes. These were generated by calculating the average per pixel of all the image nodes within each cluster. The more complex the feature vector within each node is, the slower the algorithm will run due to more comparisons required per node. Therefore, the greyscale version was also implemented, this having only one colour channel rather than three channels runs three times faster. The outcome is shown in Figure 4.4. One can notice that although there is some minor difference in the evolution of the network due to less colour information the general outcome was the same, including the number of clusters. This led us to conclude that compressing the image further down to greyscale was a feasible option. However, a similarity metric between the input and the memorised images was required to check if what is visually intuitive for humans would give the same results for a robot.

As discussed in Appendix Section A.3 various image comparison techniques are available. In this case, the metric used is the image Euclidean distance (IMED) originally proposed by Wang et al. [190] using the convolution standardizing transform (CST) as suggested by Sun and Feng [191]. Given an image with fixed size $M \times N$ written as a vector $\mathbf{x} = \{x^1, x^2, \ldots, x^{MN}\}$ the standard euclidean distance $d_E^2(x, y)$ between vectorised images x_1 and x_2 is given in Equation 4.1. This however does not take into consideration the spatial relationship between pixels.

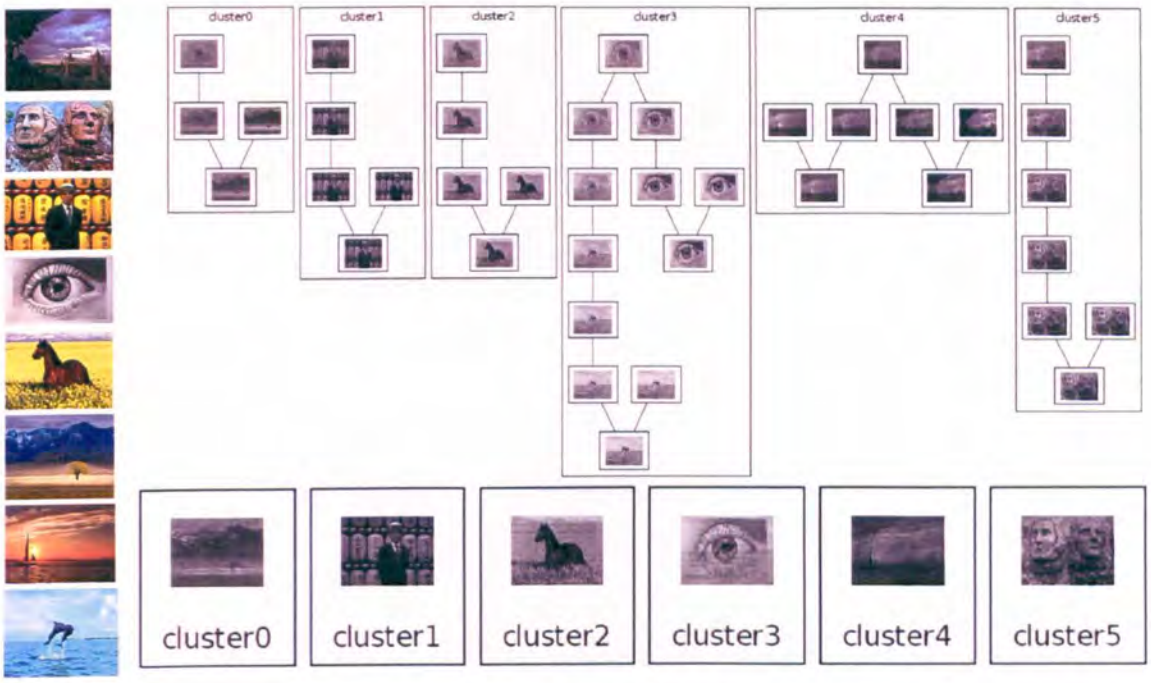


Figure 4.4: Training Images (Left), Generated GNG Clustered Nodes (Top-Right) and Keyframes (Bottom-Right)

$$d_E^2(\mathbf{x}_1, \mathbf{x}_2) = \sum_{k=1}^{MN} (x_1^k - x_2^k)^2 = (\mathbf{x}_1 - \mathbf{x}_2)^T(\mathbf{x}_1 - \mathbf{x}_2) \tag{4.1}$$

Wang et al in [190] proposed an image distance measurement which takes into consideration both the spatial relationship and the gray level relationship between pixels in images x_1 and x_2 is defined by Equation 4.2.

$$\begin{aligned} d_{IE}^2(x_1, x_2) = & \sum_{i=1}^{MN}\sum_{j=1}^{MN} g_{ij}(x_1^i - x_2^i)(x_1^j - x_2^j) \\ = & (\mathbf{x}_1 - \mathbf{x}_2)^T\mathbf{G}(\mathbf{x}_1 - \mathbf{x}_2) \end{aligned} \tag{4.2}$$

where $\mathbf{G} = (g_{ij})_{MN\times MN}$ is a symmetric and positive definite matrix. Full details together with a derivation of $\mathbf{G}$ can be found in [191]. The original method suggested by Wang et al in [190] is very computationally intensive and infeasible to use in real-time. The method proposed in [191] solves this problem by reducing the space complexity from $O(M^2N^2)$ to $O(1)$, and the time complexity from $O(M^2N^2)$ to $O(MN)$ for $M \times N$ images. The calculation of IMED between images $\mathbf{x}_1$ and $\mathbf{x}_2$ can be converted to the calculation of the traditional Euclidean distance between $\mathbf{u}_1$ and $\mathbf{u}_2$ as shown in Equation 4.3. As suggested in [191] to calculate the IMED between the input frame image and the stored memory image, both images are convolved first with the convolution template $\mathbf{K}$ (generated from $\mathbf{G}$, [191]) as in Equation 4.4 and then a standard Euclidean distance measurement is carried out on the two resulting images $\mathbf{u}_1$ and $\mathbf{u}_2$ using Equation 4.1.

$$\begin{aligned} d^2_{IE}(\mathbf{x}_1, \mathbf{x}_2) &= (\mathbf{x}_1 - \mathbf{x}_2)^T G(\mathbf{x}_1 - \mathbf{x}_2) \\ &= (\mathbf{x}_1 - \mathbf{x}_2)^T G^{1/2} G^{1/2}(\mathbf{x}_1 - \mathbf{x}_2) \\ &= (\mathbf{u}_1 - \mathbf{u}_2)^T (\mathbf{u}_1 - \mathbf{u}_2) \end{aligned} \tag{4.3}$$

$$\mathbf{U} = \mathbf{K} * \mathbf{I} \tag{4.4}$$

where $\mathbf{I}$ is the input image and

$$\mathbf{K} = 10^{-2}.\begin{pmatrix} 0 & 0.12 & 0.29 & 0.12 & 0 \\ 0.12 & 4.71 & 11.98 & 4.71 & 0.12 \\ 0.29 & 11.98 & 30.46 & 11.98 & 0.29 \\ 0.12 & 4.71 & 11.98 & 4.71 & 0.12 \\ 0 & 0.12 & 0.29 & 0.12 & 0 \end{pmatrix}$$

Tables 4.2 and 4.3 show the normalised outcome per cluster keyframe when compared to the original input images. For each keyframe the closest match to an element in the input set would be 0 and the least matching would be 1. When looking at the results shown in Table 4.2, one can see that the outcome is fairly accurate. The values in bold font show the best Input - Cluster match. Moreover, considering the eight input images and the six generated clusters, there are two keyframes (Cluster 0 and Cluster 1) which have a mix of at least two images as seen in Figure 4.3. Having said that, the closest two input images for these clusters are Inputs 8 and 2 in the case of Cluster 0 and Input 7 and Input 2 for Cluster 1. However Input 2 has a closer match in Cluster 0. This shows that even though the Keyframe has interlaced images in it, it is still a good tool to identify and cluster input images after training. Results for the greyscale version shown in Tables 4.3 provide similar, if not, somewhat better results, thus reinforcing the original assumption that using greyscale images will give us good results for the application.

4.3 Building visual memories of video streams

Taking inspiration from video segmentation techniques, this section looks at an efficient method for learning and memorizing an environment in real-time from a sequential input video stream in a very concise and compact manner, primarily intended for robot understanding of its own environment and possibly localisation. Various video segmentation methods exist in literature [192, 172, 193, 194, 195, 196, 197, 198] and some are extensively reviewed [199, 200, 201, 202, 203, 198]. The generally accepted content-

	Inputs	Cluster0	Cluster1	Cluster2	Cluster3	Cluster4	Cluster5
1.		0.7115	0.6904	**0.0000**	0.9437	0.9929	0.6198
2.		**0.2233**	0.3473	0.6953	0.6567	0.5744	0.6307
3.		1.0000	0.8041	0.7510	0.7634	1.0000	**0.0000**
4.		0.4953	0.6521	1.0000	0.8112	**0.0000**	0.8160
5.		0.6848	0.5533	0.9083	**0.0000**	0.7525	0.6250
6.		0.5892	**0.3626**	0.7110	0.8017	0.5509	0.6773
7.		0.4992	**0.0000**	0.5915	0.4606	0.6150	0.4767
8.		**0.0000**	1.0000	0.9814	1.0000	0.7161	1.0000

Table 4.2: Normalised similarity score - colour (0-max, 1-min)

	Inputs	Cluster0	Cluster1	Cluster2	Cluster3	Cluster4	Cluster5
1.		1.0000	0.4832	1.0000	1.0000	**0.0789**	0.8964
2.		0.5636	0.5643	0.4961	0.4248	0.3015	**0.0000**
3.		0.9764	**0.0000**	0.7708	0.7931	0.5493	0.8632
4.		0.7273	1.0000	0.6685	**0.0000**	1.0000	1.0000
5.		0.8912	0.9346	**0.0000**	0.3404	0.8885	0.9260
6.		**0.0000**	0.6595	0.6594	0.5422	0.6397	0.8427
7.		0.5575	0.5242	0.5247	0.3254	**0.0000**	0.5017
8.		0.8728	0.9695	0.5223	**0.1393**	0.7458	0.8221

Table 4.3: Normalised similarity score - greyscale (0-max, 1-min)

based video segmentation approach is to first break up the video sequences into temporal homogeneous segments called shots, then to condense these segments into one or a few representative frames commonly referred to as keyframes, and finally to organise similar shots based on the keyframes or other audiovisual characteristics to construct a compact and hierarchical representation of video for browsing and retrieval [172].

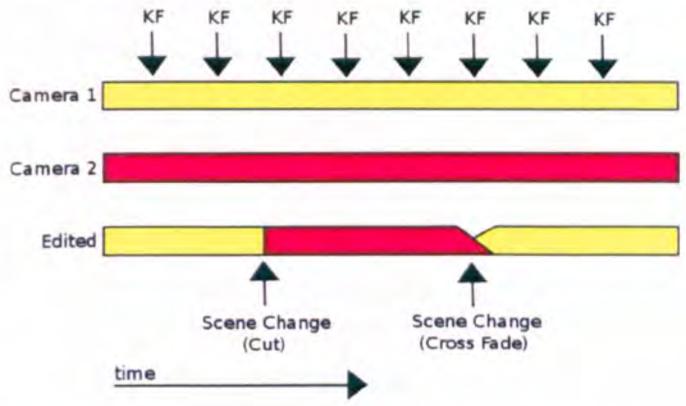


Figure 4.5: Keyframes and scene changes in video segmentation

Figure 4.5 provides a graphical description of keyframes (abbreviated as KF) and scene changes. Having an edited video from two cameras, the two most common scene changes between cameras are generally cuts and cross fades. In cuts the change is abrupt whereas in cross fades the transition is carried out over a number of frames with one scene is fading out whilst the other is fading in. The simplest way of summarizing a video sequence is that of grabbing a keyframe every X number of frames. Another more elaborate way of providing keyframes describing various scene changes is to first detect scene changes or change of events in a video sequence and then the sequence is traversed again to select best keyframe representing each extracted shot [195]. The main limitation of such methods generally is that they cannot segment and annotate in real-time and assume that the video is stored and can be post-processed. Moreover, some algorithms can also be very computationally intensive [203]. Chatzigiorgaki and Skodras [195] suggest a method for real-time keyframe extraction based on a sequential search algorithm that bypases the process of temporal video segmentation. Ngan and Li in [203] highlight four main challenges in image/video segmentation. The first challenge is how to bridge effectively the semantic gap between low-level and high-level features. The second is how to yield accurate segmentation and how to extract accurate masks. The third challenge is that of working in real-time without compromising accuracy and the fourth is the need to develop appropriate validation and evaluation approaches, by providing a common database and by developing an evaluation technique. Ngan and Li state that most evaluation methods in the current literature are based on the computation of the scores between the ground truth mask and the segmented result. It is reasonable but not sufficient to address the segmentation quality. Gao et al. [202] state that the usage of machine learning techniques has proven to be a robust methodology for semantic scene analysis and understanding. As neural networks are capable of learning the characteristics of various video segments and clustering them accordingly, in this section, a neural network based technique is developed to segment the video sequence into shots automatically and with a minimum number of user-defined parameters [172].

As highlighted in Section 4.2.1, GNG has the main advantages of growing over time, cater for adaptive clustering and automatically determine the number of clusters based on the characteristics of the data. The way GNG is used in this study bears some resemblance to well-known video annotation and segmentation methods [203] and [204], however with the main advantage of having segmentation evolve automatically in real-time. Video segmentation techniques generally look for specific changes in the video segment such as scene change or cross fading between scenes. They also use more complex segmentation algorithms such as graph-cuts and eigen-based methods. Moreover, in video segmentation, a keyframe is generally defined as a frame in the time line where something new appears. Also, such algorithm would be interested in extracting specific frames in the time-line. In this case, rather than just locating a new appearance in time, the main aim is to extract knowledge from the visual input and store it compactly in a way that both humans and robots can use it to gain knowledge of their environment. Using GNG each incoming frame contributes to evolve the visual memory as opposed to just grabbing specific frames as done in video segmentation. This would evolve in a clustered graph with nodes representing similar scenes. The generated topological graph will link nodes based on similarity rather than temporality. Therefore, for this application, a keyframe can represent more than one frame, compacted together and yet still representative of the input visual information as shown in Section 4.2.3.

4.3.1 Visual memories using GNG

In robot video streams, cuts are generally not expected, but rather a lengthy continuous video sequence is provided. This is even more so if robots are not sharing information and each robot is working independently with no input coming from other sources. Scene changes therefore can happen over a very wide span of frames. This necessitates the need of a continuously adaptable topology. It is therefore suggested that rather than storing images when there is a drastic change as suggested in [165] and other works mentioned earlier in Section 4.2 the proposed method adapts images within memory through a set of evolving nodes within a network. Following from the biological inspiration described in earlier sections, the proposed algorithm does not produce a perfect photographic memory, but rather retains image representations, which contain meaningful information about the explored environment. This choice provides several advantages when implemented for a robot. Storing and transmitting a video stream requires a large amount of memory and a high bandwidth, usually scarce on a robot. For the robot application, feeding a random sample from a stored sequence (as is generally done in unsupervised training techniques) is not possible.

A system which is capable to learn, adapt its knowledge, accept a continuous video stream and process it in real-time is required. Kirstein et al. [66] state that the GNG was originally designed for offline training, therefore an adaptation for the GNG to be capable of accepting video streams was needed. In [23] the case of using GNG on a robot streaming sequential images from its environment was studied and a solution for sequential streaming as opposed to standard neural network training techniques was provided.

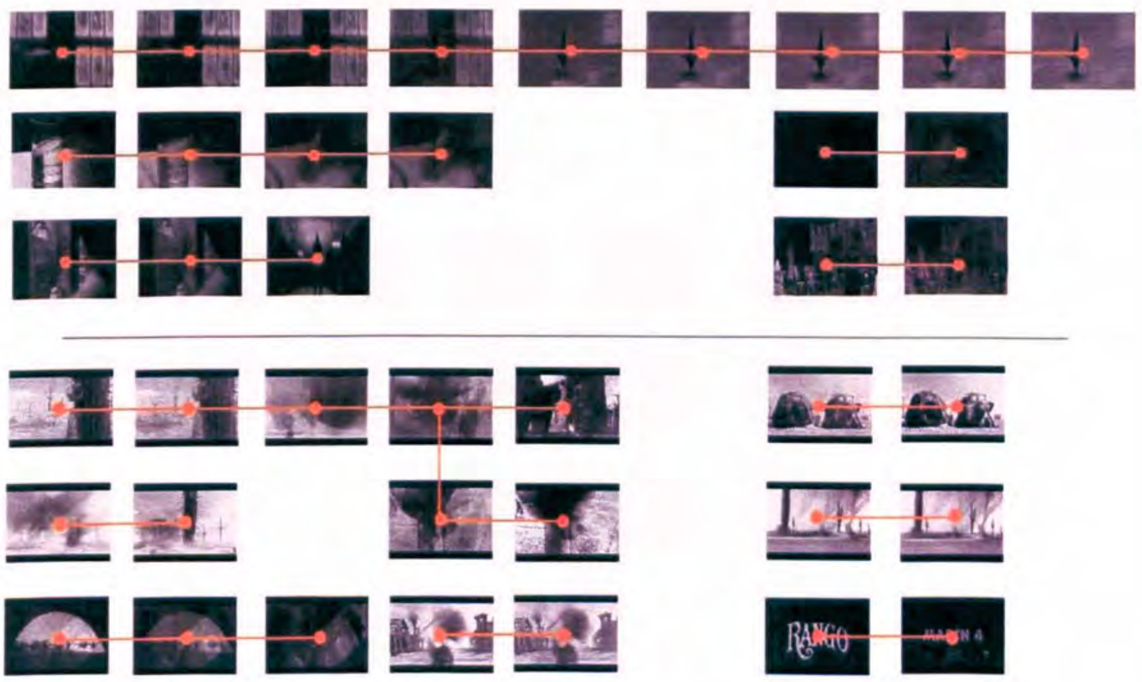


Figure 4.6: Generated images from movie trailers (Inception and Rango), automatically memorised and sorted into clusters.

GNG's performance was initially tested with the standard setup and video sequences available on the internet and sequences captured by the robot cameras. For these experiments, a new node was introduced in the network every 50 frames up to a maximum of 20 nodes. Nodes which become isolated are removed. Figure 4.6 shows the outcome obtained when feeding movie trailers to the GNG. This was then repeated using an input stream from the robot camera which was set up to capture a video of the laboratory. Figure 4.7 shows the outcome for two of the memorised images (right) alongside an original frame image (left). Figures 4.8 and 4.9 show the difference in applying GNG directly on two video streams, one which is smooth and the other which

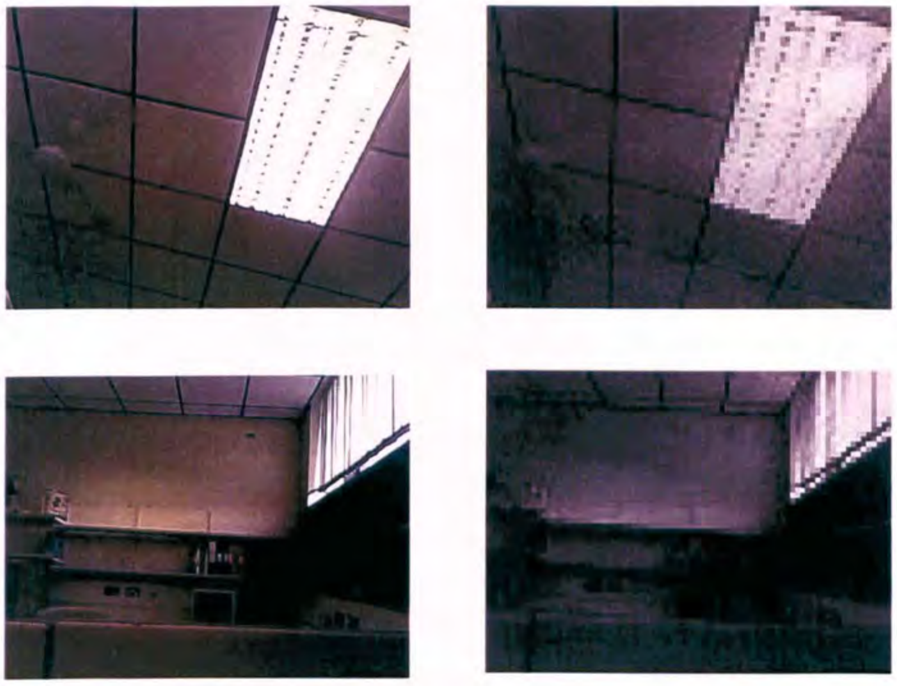


Figure 4.7: Original frame image on the left and memorised frame image on the right

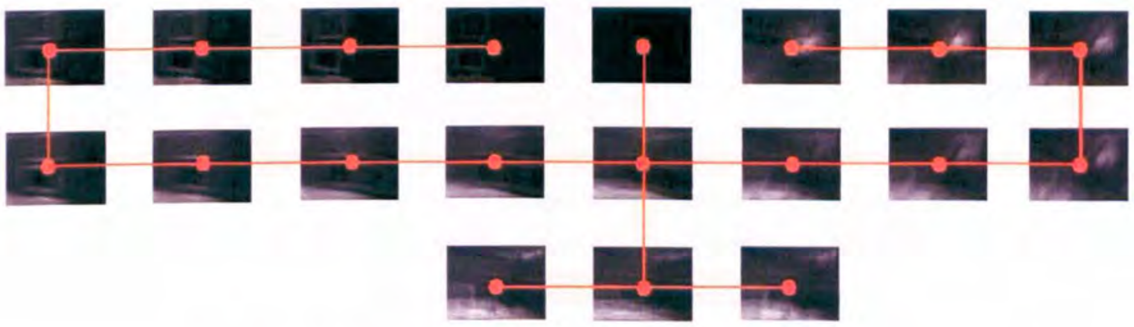


Figure 4.8: Clusters formed from a smooth moving camera

is moved swiftly and stopped at fixation points. The smooth movement was intended to represent the conventional way of filming or grabbing a video. The movement of the camera was in a way to mimic human fixations, basically looking at a place and locking on it for some time if it is of interest, looking randomly somewhere else, again locking if seeing something of interest, looking back to original pose and looking at random when nothing of interest is found. This could, in a way, be considered as an "edited" video or a movie clip with several shots taken from various angles. Again, as done in the filming industry, having suddenly changing shots grabs attention. The results obtained by moving the camera smoothly versus rapidly with very distinct looking directions were as expected. A maximum number of 20 nodes were created in the GNG. The smooth moving camera GNG output resulted in the 20 nodes of very similar consecutive images that are all merged into a single cluster as shown in Figures 4.8. The GNG

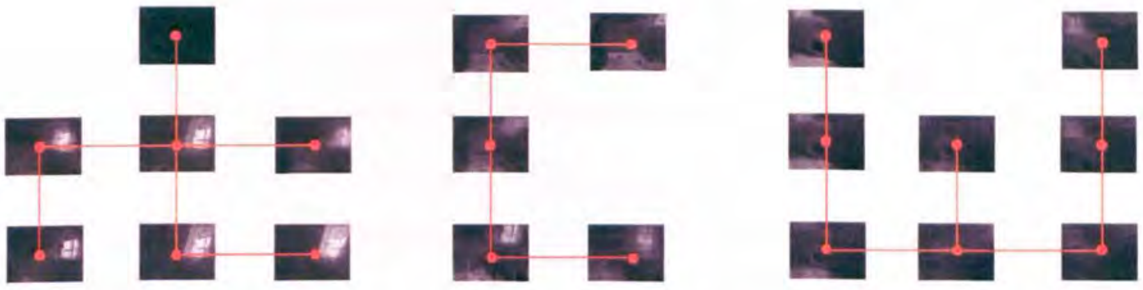


Figure 4.9: Clusters formed using a random / fixation moving camera

fed with frames from the camera sweep with a more random movement evolved into 3 clusters as shown in Figure 4.9. It can be noticed that when a video sweep comes from a smooth moving camera, due to the adaptability of the GNG, the nodes have time to slowly adjust and become relatively similar in one big cluster. However, as the movements become more rapid, so does the element of randomness. This leads to the creation of very distinct consecutive images leading to better clustering performance.

The GNG would work relatively well if the camera was affixed on a person's head(e.g. camera on security guard). Even if this solution was implemented as is, as a possible solution, having a robot continuously looking at random at the environment is not ideal for various reasons. Extra hardware and interfaces would be required to rotate the camera. Also, this would lead to more mechanical movement implying faster power depletion, which is a very important and limited resource in a robot. Due to the sequential nature of the robot's visual data acquisition, some modifications were required on the GNG learning mechanism over the proposed methodology indicated in Section 4.2.1. This was initially tested on movie trailers and then onto actual robot video streams as will be further seen in Section 4.3.3:

- As also highlighted in [65], during testing it was noted that the best matching unit (BMU) is the same one for a number of consecutive frames which are very similar. In this case it is good, however one needs to be careful not to overtrain. If this happens one possible option could be that of skipping input frames.

- A fast learning rate is also required. The BMU is made to converge to the actual input very quickly by adding a large proportion of the error between the input and the BMU. This basically sets the BMU to the input image.

- The second BMU (for the same frame) is slightly adjusted. This means that whereas the BMU is made to converge very fast (+90% of error), the second BMU is only made to converge minimally (+1% of error). This allows for more information storage within the same cluster, rather than having several nodes with the same value within the cluster.

- A new node is inserted at a relatively fast rate (e.g. every other iteration). Thus allowing for a large number of nodes to be used from early age and new nodes are inserted soon after isolated nodes are removed.
- For the short term memory case (Described in Section 4.3.4), an extra step to remove 'old' clusters whose nodes do not contribute to recent information was included.

4.3.2 Parameter tuning

Several parameters need to be set in GNG. Although the way the input video is designed (e.g. the number of cuts and camera changes present) and the processing power available on the robot influence the parameter selection, a general trend for such selection can still be defined as will be described next. A good compromise between the number of meaningful formed clusters (keyframes) and performance time was required. The most crucial factors affecting the graph real-time performance are the maximum number of allowed nodes, the insertion rate of new nodes, and the maximum age allowed for edges between evolving nodes. The more nodes present in GNG, the more processing intensive the algorithm becomes. For this application, the parameter and maximum node number selection criteria consist of the ones resulting the largest number of meaningful non-repeated keyframes obtained in real-time on the robot. To select the parameters a video sequence[1] suggested in [195] was used. An initial experiment showed that an edge age greater than 20 would result in one big cluster. On the other hand, a too small number would remove nodes too quickly. For this video sequence, the edge age was set to 4.

	Maximum number of nodes							
New node every X input frames	10	20	30	40	**50**	100	200	500
2	3	3	5	6	7	12	14	16
4	3	4	8	8	8	10	11	11
8*	3	4	5	7	6	8	$8^{(110)}$	$8^{(110)}$
16	4	7	8	9	**9**	$9^{(52)}$	$9^{(52)}$	$9^{(52)}$
32	2	$2^{(13)}$	$2^{(13)}$	$2^{(13)}$	$2^{(13)}$	$2^{(13)}$	$2^{(13)}$	$2^{(13)}$
* processing time (sec.)	16	20	25	27	**28**	45	51	51

Table 4.4: Parameter selection (Numbers in brackets indicate the number of nodes reached when the number of final nodes is less than maximum number of nodes allowed.)

Table 4.4 was generated by keeping gain coefficients constant and a maximum edge age

[1]http://www.youtube.com/watch?v=mSkP43A-LQ4

set to 4. The insertion of new frames was varied at powers of base 2 and run over GNG with set maximum number of nodes ranging between 10 and 500. Gain coefficients for the best and second best matching units were set to 0.3 and 0.003 respectively and β was set to 0.005. To find out the best combination in relation to time, the video images are made available in a sequential manner and the GNG algorithm processes them as quickly as the robot processor allows. The robot processor used was an Intel® Core^{TM}2 Duo CPU T8100 @ 2.10GHz $\times$ 2, 3.9GB Memory running Ubuntu 12.04 32 bit. The numbers in brackets indicate the number of nodes populating the network by the end of the video sequence. These are only displayed when this value is less than the maximum number of allowed nodes. The video sequence is 30 seconds long, when run at 30 frames per second. This means, that given the processor on the robot, the best combination for real-time operation / useful cluster outcome would be:

- number of nodes: 50,
- new node every 16 input frames, and
- an edge age of 4.

Allowing more than 50 nodes, each consisting of 80×60 RGB values, would compromise real-time operation. The GNG graph and keyframes generated using the above settings are shown in Figure 4.10.

The nine generated visual memory keyframes were then compared to the 25 MPEG-B-frames with the values provided in Table 4.5. It can be noted that most MPEG keyframes match the visual memory keyframes with a very high score ($\approx$ 0.0). The worst match occurs on MPEG keyframe 15, being most similar to Cluster 7. In this case, the best matching score obtained is that of 0.4616 which is much higher than the ideal expected 0.0 value. Intuitively, one would rather match this keyframe to Cluster 0. Although from the values obtained, Cluster 7 appears to be the least performing, one has to note that in the MPEG frames, no keyframe appears to visually match Cluster 7.

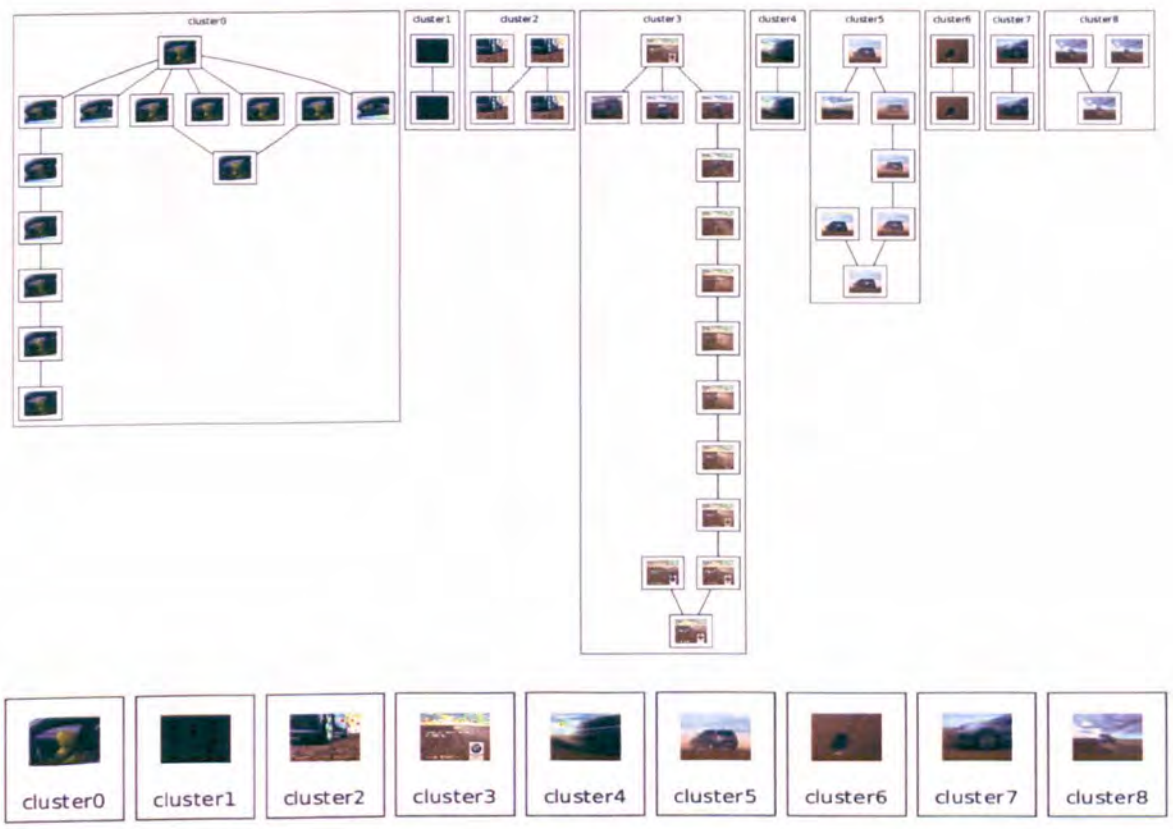


Figure 4.10: Generated GNG graph and corresponding keyframes

Table 4.5: Normalised similarity score - MPEG Keyframes (0-max, 1-min)

		Cluster0	**Cluster1**	**Cluster2**	**Cluster3**	**Cluster4**	**Cluster5**	**Cluster6**	**Cluster7**	**Cluster8**
	MPEG KF									
1.		0.4258	**0.0000**	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
2.		0.5489	0.5398	0.5275	0.3646	0.1659	0.3674	**0.0000**	0.0550	0.3082
3.		0.8760	0.8852	0.8615	0.4283	**0.1903**	0.2676	0.8348	0.2277	0.2380
4.		0.8745	0.9042	0.7279	0.2902	0.2937	0.2210	0.7726	**0.0607**	0.1735
5.		1.0000	1.0000	0.8512	0.3320	0.3986	**0.2072**	0.9514	0.2038	0.2452
6.		0.5649	0.5514	0.5577	0.3833	0.2181	0.3807	**0.0726**	0.1239	0.3268
7.		0.9058	0.9180	0.7138	0.2739	0.4871	0.2396	0.7994	**0.1272**	0.2059
8.		0.8688	0.8974	0.7189	0.2738	0.2925	0.1619	0.6722	**0.0455**	0.1376
9.		0.6266	0.6342	0.6271	0.3499	0.0426	0.3179	0.1175	**0.0000**	0.2718
10.		0.8123	0.8376	0.7359	0.3004	0.0829	0.1930	0.5921	0.0209	**0.0000**

Continued on next page

Table 4.5 – continued from previous page

	MPEG KF	Cluster0	Cluster1	Cluster2	Cluster3	Cluster4	Cluster5	Cluster6	Cluster7	Cluster8
11.		0.8065	0.7939	0.7811	0.4039	**0.0000**	0.2779	0.7044	0.0052	0.2772
12.		0.9082	0.9357	0.8091	0.2732	0.1433	0.1099	0.7339	**0.0557**	0.1336
13.		0.9831	0.9915	0.8217	0.3367	0.4159	**0.0000**	0.8802	0.1548	0.2648
14.		0.7925	0.7747	0.8016	0.5246	0.4246	0.4867	0.7393	**0.2045**	0.5065
15.		0.5333	0.5862	0.9025	0.7088	0.4915	0.7078	0.7918	**0.4616**	0.7042
16.		**0.1725**	0.3489	0.8846	0.7631	0.4848	0.7604	0.6811	0.5512	0.7243
17.		0.7178	0.6579	**0.0000**	0.4925	0.7882	0.5025	0.6150	0.4064	0.5314
18.		0.7165	0.6562	**0.0060**	0.4922	0.7820	0.5023	0.6134	0.4047	0.5314
19.		**0.1355**	0.3084	0.8592	0.7601	0.5122	0.7594	0.6532	0.5647	0.7194
20.		**0.0000**	0.3254	0.8346	0.7479	0.5571	0.7576	0.6574	0.5482	0.7291
21.		0.8047	0.8006	0.7086	0.2680	0.2562	0.3023	0.6461	**0.0231**	0.2971

Continued on next page

Table 4.5 – continued from previous page

		Cluster0	Cluster1	Cluster2	Cluster3	Cluster4	Cluster5	Cluster6	Cluster7	Cluster8
	MPEG KF									
22.		0.8771	0.9120	0.7133	**0.1287**	0.5160	0.2552	0.7069	0.1547	0.2571
23.		0.8867	0.9056	0.6965	**0.0508**	0.4654	0.1972	0.6906	0.1173	0.1982
24.		0.9037	0.9343	0.7343	**0.0000**	0.5464	0.2680	0.7812	0.1955	0.3033
25.		0.8983	0.9328	0.7388	**0.0265**	0.5360	0.2761	0.7804	0.1896	0.3131

4.3.3 Robot video streams

As suggested in Section 4.3 movie trailers have scene and camera changes, usually abrupt, which would generally imply that a new cluster representing a new scene will start evolving within the network. However, inputs from robot cameras do not have such changes and a way to test the above method was needed. Video segmentation is application oriented so it is very difficult to measure a given segmentation quality based on a uniform criteria [203].

Following the results obtained in Section 4.3.2, in this section the practicality of the method proposed is tested to see how it compares to a real life robot situation by feeding a video sequence obtained from a robot platform navigating through a corridor. An indoor navigation and localisation testing dataset available on youtube[2]. This consists of a camera fixed on a mobile robot moving along corridors. This video stream was fed in the algorithm consisting of a maximum of 50 nodes. The 50 generated nodes were automatically divided into four clusters with Figure 4.11 showing the four cluster keyframes.

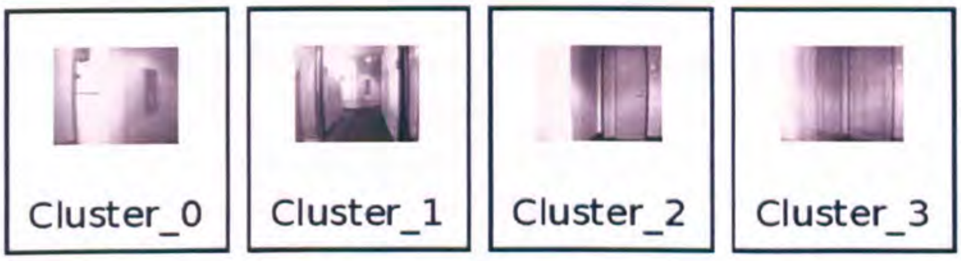


Figure 4.11: Corridor Keyframes

The next step involved the use of these these keyframes by the robot for localisation purposes. A check, based on the comparison principle described in Section 4.2.3, on whether the robot is capable of identifying in which region of the corridor it finds itself in is carried out. In this case, the testing and training data come from the same video source, however, since frames are skipped to avoid overtraining (as suggested in Section 4.2.1) such skipped frames can be employed for testing. Moreover, it is assumed that the robot will navigate along the corridors in a relatively similar way each time. Figure 4.12 shows the outcome. In this plot, the x-axis represents the frame number of the input video stream and the y-axis represents the comparison value between the input frame and the matching keyframe. Each input frame was compared to all four keyframes and the Best Matching Unit (BMU) was noted. The proposed methodology successfully manages to identify the best keyframe based on the current robot position. Although the ground truth can be relatively fuzzy in this case (due to the humans' subjective behaviour in stating consistently in which part of the corridor robot is [203])

[2]http://www.youtube.com/watch?v=L3RiF-VASc4

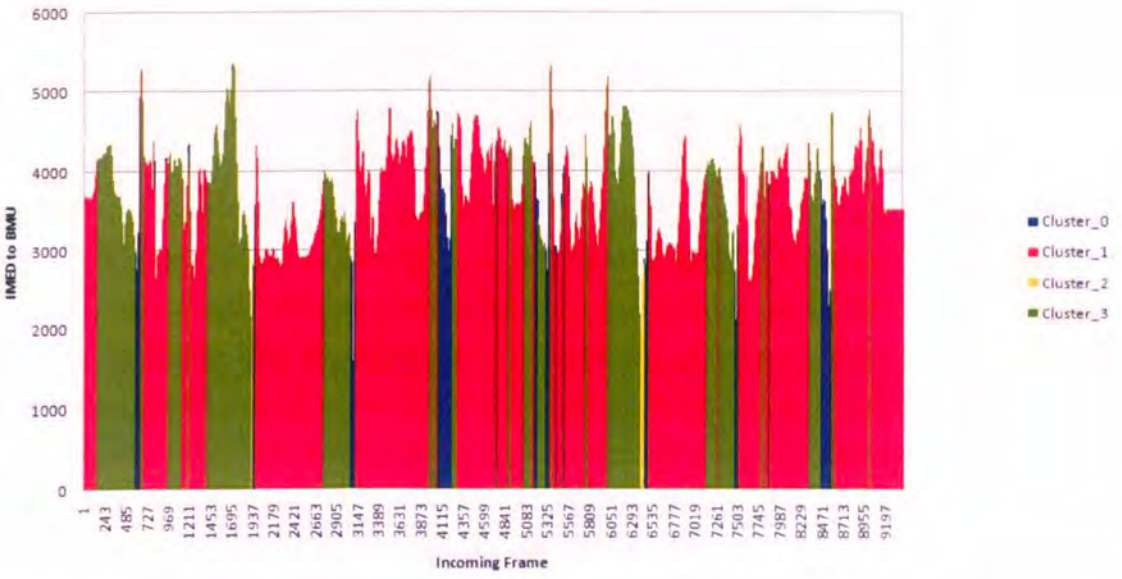


Figure 4.12: Comparison of a node with a new incoming video stream. The lower the value, the more similar the incoming image is to the memorised image.

Figure 4.12 shows that the BMU (selected keyframe) for the input frame matches are, more often than not, correct. This is further reinforced by the fact that the matches are consistent for consecutive frames which is the case for a robot navigating around corridors. This allows for outliers to be easily identified and possibly corrected.

4.3.4 Temporal memory

As already discussed in Section 4.1, human memory can be divided into short term and long term memories (STM and LTM). This section proposes a method which endows robots with (such) memory capabilities. Moreover, a time factor to when the memorised scene took place was incorporated, again stored in a compact representation.

In this implementation GNG is used to generate the STM from which information is extracted to feed a linked list populating the LTM. The Temporal LTM would give us a rough occurrence indication of the scene relative to an initial storage phase. STM should only contain recent memory. One way of maintaining STM is to remove nodes (which are) not contributing to recent information. This means that nodes which have been static for a long period of time are removed from the network. These free nodes can thus be inserted in a more dynamic cluster of the network to reflect better the incoming visual stream. The information within the 'old' nodes however must be retained. This is where LTM comes in. For LTM, a snapshot of the clusters is taken and stored as a keyframe. The mean of the cluster is calculated and the output is one resulting image. Some inference on the importance of cluster content can also be inferred from its age and size when generating keyframes. A large cluster would generally indicate that there are several nodes holding similar information and that the cluster is relatively young. Each time a node is updated, its age is reset to 0. Therefore, it is expected that reoccurring scenes will remain in the STM. On the other hand, less frequently occurring scenes stored in STM nodes will stop being updated, however their age will keep increasing. Since similar nodes will form their own cluster, the overall cluster age would represent how 'old' the information within that cluster is. It is argued that clusters with an average age above a certain threshold could be classified as containing static data and therefore its contents can be transferred to LTM, leaving space in the STM for new, more dynamic information storage. Figure 4.13 shows some of the clusters obtained when feeding a movie trailer (Up in the air)[3] to the visual memory as suggested in Section 4.3: the subset of the generated clusters and the keyframes obtained as their means. These keyframes further reduce the information used to generate the LTM images. A video of the evolving GNG can be watched on youtube[4]. Figure 4.13 merits some comments:

- **Cluster_0**: The cluster contains two virtually identical images and in this case the keyframe has a very sharp image stored.
- **Cluster_1**: The two images in the nodes represent different scenes, however, they are very similar. The keyframe in this case is somewhat blurred where the actors appear however there is still a good understanding of the scene.
- **Cluster_2**: In this case, the cluster contains very sharp images of scenes of the movie trailer however this provides us with a blurred keyframe. Still, some information can be inferred from this. By time, links in the cluster will start breaking up due to the age constraint on the edges.
- **Cluster_3**: is the youngest cluster with lots of similar nodes. this again provides a sharp mean image.
- It can therefore be deduced that Cluster_0 and Cluster_1 are older than Cluster_2 and Cluster_3.

As shown in Algorithm 2 the temporal LTM works as follows. At every X number of iterations the oldest cluster within the GNG is identified and its average image taken. The cluster age is calculated by summing up and averaging the age of each neuron within its cluster. This is compared to existing images within the LTM. If a very similar image is found the same node within the LTM is updated and the last node in LTM is linked to this most similar node. If not a new node will be created and

[3]http://www.youtube.com/watch?v=e7k6FwXJhNk
[4]http://youtu.be/vlSLJWQvl5M

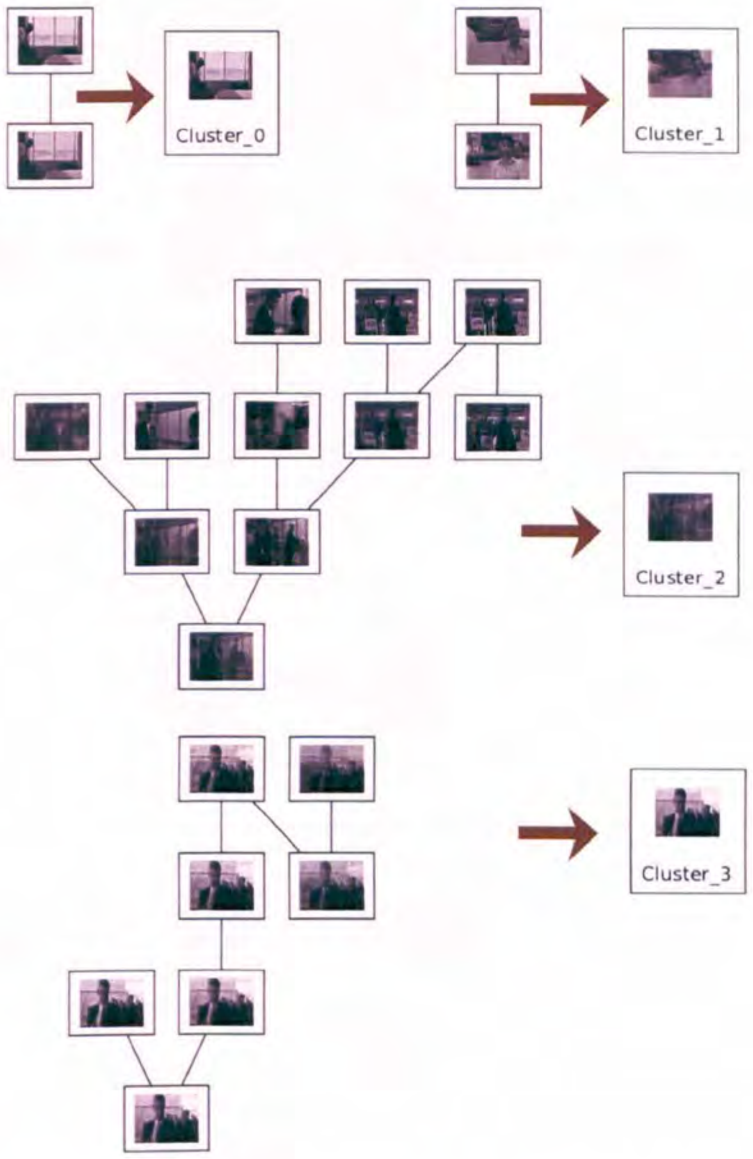


Figure 4.13: Keyframes of the Generated Clusters

linked to the last node. The temporal link would give an indication on how clusters were formed in time. The oldest cluster and all its nodes are deleted from the GNG (STM) thus allowing for new data to be learned at a much faster rate, given that many more nodes are available. Figure 4.14 shows the resulting LTM whilst the most recent information remains within the STM as seen in Figure 4.15.

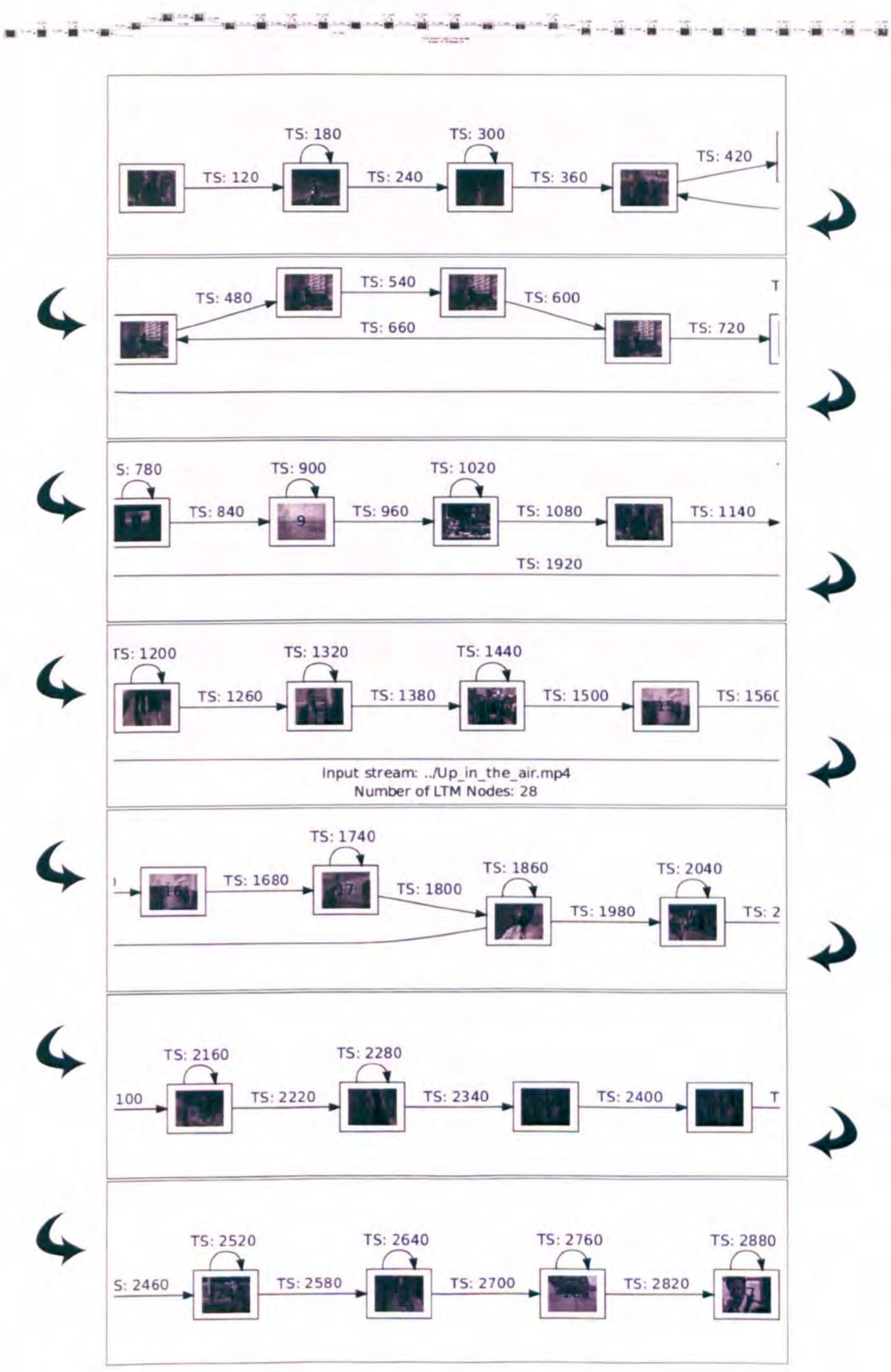


Figure 4.14: Temporal Long Term Memory

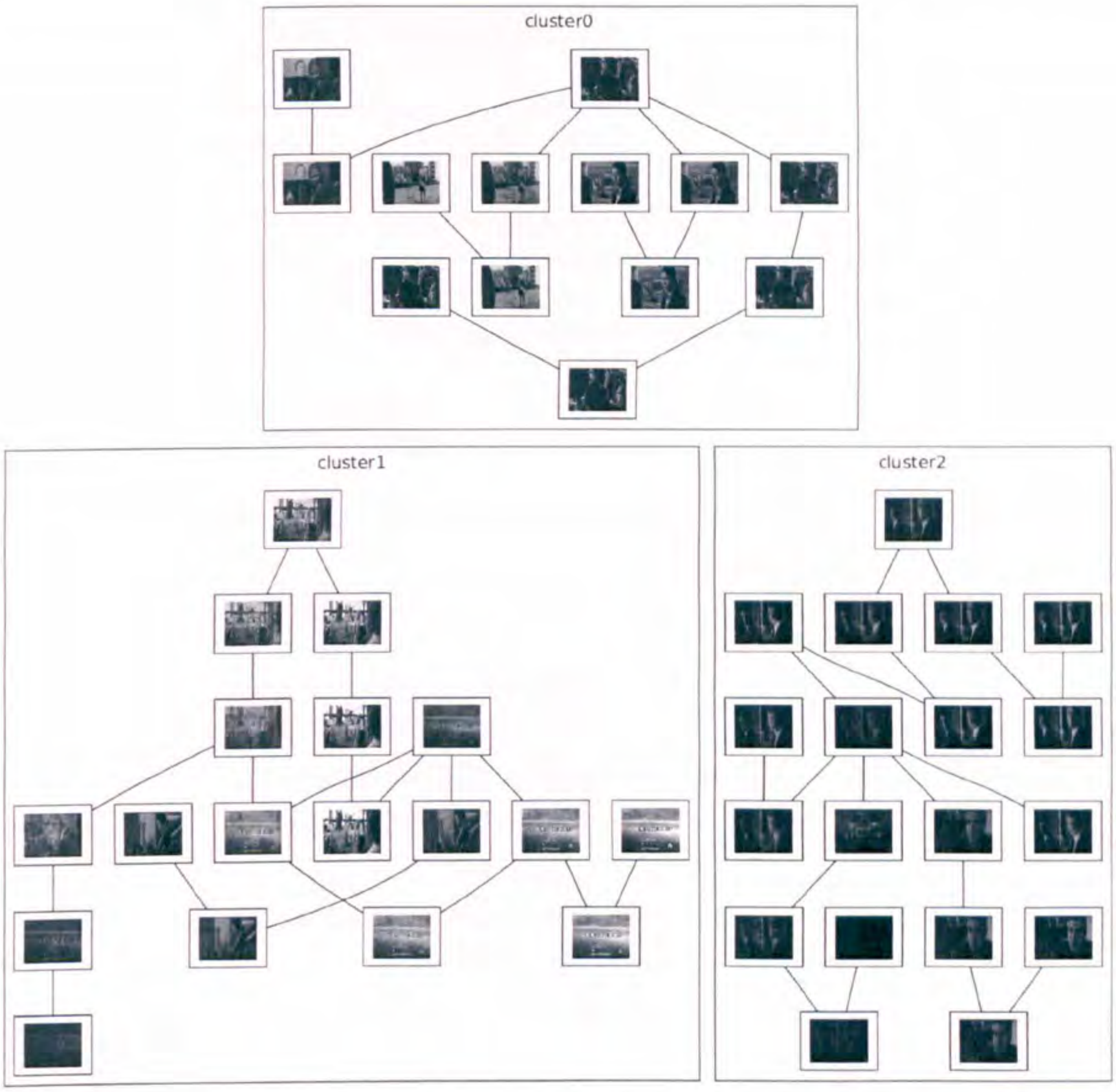


Figure 4.15: Short Term Memory

4.4 Summary

This chapter provided a method where robots efficiently learn and memorize their environment in real-time from a sequential input video stream into a flexible graphical representation using a Growing Neural Gas (GNG) network. The system was tested on various video streams, both manufactured (movie trailers) and raw (from robot). Experimental results show that the proposed method suits its intended application and a very concise yet meaningful representation of input data is obtained. The proposed system is fed with the scaled down raw images. Whilst carrying out experiments it was noted that if the movement of the robot is not smooth, sequence frames capturing the same scene will generate a different Euclidean distance thus leading to the tendency of generating multiple clusters of the same scene which although not critical can be a nuisance at times and leaves space for further algorithm enhancement. One possible

Algorithm 2 Generating Temporal LTM

```
while (Stopping Criterion = false) do
  if (current iteration is a multiple of X) then
    - Identify oldest cluster within the STM
    - Produce cluster average image (keyframe)
    - Compare to existing images within the LTM
    if (very similar image node found within the LTM = true) then
      - most similar LTM node is updated
      - Last LTM node is linked to this most similar node
    else
      - A new node is created
      - Linked to the last node
    end if
    - Delete oldest cluster and all its nodes from STM
  end if
end while
```

way to overcome this is by using features which are position invariant and video stabilisation techniques such as those suggested in [9]. This however might compromise the real-time capability and also leads to a higher processing time due to more complex computations on the incoming video stream.

As will be seen in Chapter 5, this work is also extended to a team of robots collaborating together and sharing visual memories focusing on the capability of robots learning from each other with the final aim of having robots working together as a team for environment monitoring and learning.

"*We are drowning in information but starved for knowledge.*"

John Naisbitt, *American author and public speaker (1929 -)*

Chapter highlights:
Combining work presented in earlier chapters, this chapter presents an actual multi-robot platform implementation in which robots operate in a modular fashion, sharing and merging information, thus creating a more holistic picture of their dynamic environment.

5 Multi-Robot Information Fusion

As working environments become larger and tasks to be performed become more complex, the problem of capability limitation of a single robot becomes more obvious [205], hence the need for multi-robot interaction. This requires robots to be able to share information coming from various sources. When data are shared, even though it is not exactly the same, it provides a useful starting point - a prior [28]. In addition, as robots continue to perform their tasks and gather their data, the quality of prior information improves and begins to reveal the underlying patterns and correlations about the deployed robots and their environments [28]. Ultimately, the nuanced and complicated nature of human environments cannot be synthesized with a limited set of specifications, but requires robots to systematically share data and build on shared experience [28].

The previous chapters described how robots can be used to perceive and gain knowledge about their environment. This chapter presents methods to make robots operate, share and combine information successfully, within their environment, based on multi-robot system architectures, coordination and cooperation previously discussed in Section 2.5. Whilst keeping modularity of sensors, actuators and processes, both on the robots themselves and also those scattered within the environment, information fusion techniques were studied and development frameworks are analysed. Methods on how all the acquired information is fused so as to obtain a more accurate and global picture of the environment and also methodologies to filter out the necessary information and avoiding information overload is presented.

5.1 Robot platform integration

Based on the application, described in Section 1.2.1, security operators are increasingly interested in solutions that can provide an automatic understanding of potentially crowded public environments [206]. Robots are only one set of links in a bigger chain. There is therefore the need to create a heterogeneous intelligent multi-sensor platform, combined over a multi-robot / multi-agent scenario, so as to have smart monitoring of complex public scenes to monitor, understand and interpret complex public environments consisting of:

- human security guards equipped with sensors and smart devices,
- mobile robotic platforms, and
- fixed cameras.

From the robots' perspective, robots should be able to correctly perceive their surroundings so as to successfully operate within this highly dynamic environment; and this had to be taken into consideration whilst setting up the robots. Based on the multi-robot platform introduced in Section 1.3, seen in Figure 5.1 (same as Figure 1.3 but repeated for clarity), a practical, robust and efficient solution for the actual implementation of this application was required. Solutions for intelligent integration of sensorial information coming from different sources together with effective human-robot interaction methods within the multi-human, multi-robot paradigm are required. Moreover, an effective way of coordination among robots, vision sensors and human guards is also required. As already mentioned in Section 1.1, robots should be able to operate in their environment, whilst extracting information of interest, both automatically and on demand, by:

- operating in a top-down fashion through a central directive, and/or
- trigger bottom-up operation prompted by information extracted and analysed from individual robots.

One of the main challenges in setting up the robots consists of having a complete platform which operates in real-time, scalable over multiple robot platforms and also capable to communicate with other external systems. The various components available on the robots are usually controlled by software modules developed by different manufacturers using different programming languages. These components may also use different communication mechanisms. Software modules are also needed to process sensor information and control actuators to perform computational, vision and cognitive tasks like planning, navigation and user interaction. As already stated, complete

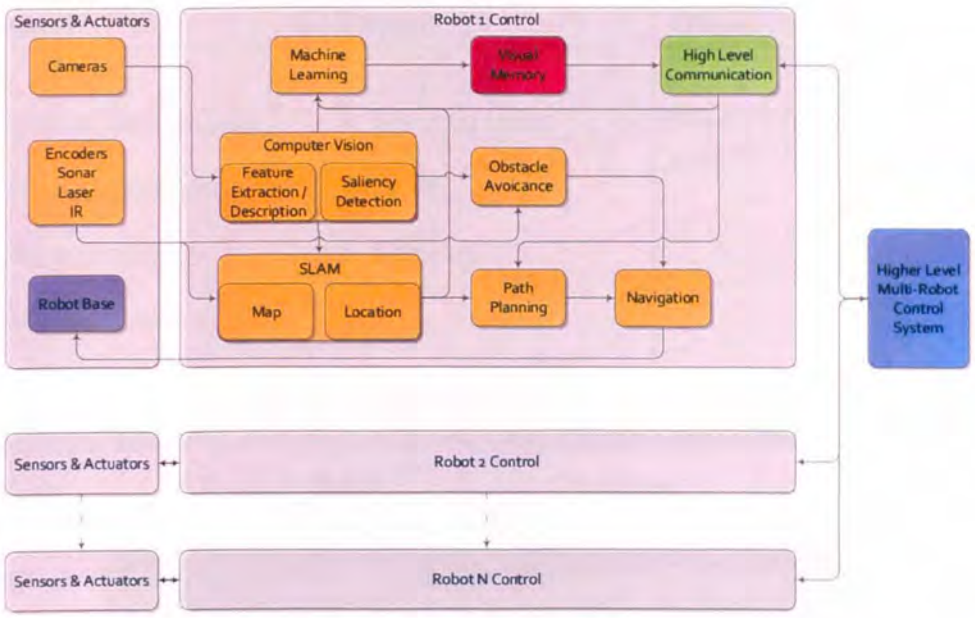


Figure 5.1: General Overview of a Multi-Robot Platform (same as Figure 1.3)

integration for real-time operation is of utmost importance. The right choice for the best suited development platform is still considered as one of the major bottlenecks in large scale multi-robot design especially due to the lack of set standards [207].

In the real world not all hardware/software is developed on the same framework and generally one only has control over one's own developments. The proposed methodology allows for the integration of tailor made developments together with other already existing frameworks, which as generally happens, are not necessarily working on the same framework. As in this case, it was necessary to interface software working on remote machines having different operating systems, namely Linux, Android (tablet) and Windows. In this case data was transferred using TCP/IP protocol over Ethernet.

Table 5.1 shows the main components of each robot platform required to operate for the desired application, as described in Section 1.2.1, implemented at Kingston Hill Campus in August 2012. Each robot platform consists of a hardware architecture, a software development framework and additional equipment required to interface with other existing agents in the environment. The hardware architecture is further described in Sections 5.1.1. Abstract Section A.6.4 highlights various software development and middleware frameworks available, from which the preferred methodology for the robot platform was then selected and presented. Moreover, further details on the required setup for the robots can be found in Appendix D.

5.1.1 Hardware architecture

In classical robot design, robots were dedicated machines designed to achieve specific tasks and manufactured as one single unit [208]. As robot platforms became larger,

Hardware	Software	Other
• Erratic base robots • Kinect cameras • PC webcams • laser scanners • laptop	• Ubuntu linux • ROS • Opencv	Wi-Fi communication over TCP/IP

Table 5.1: Robot Platform

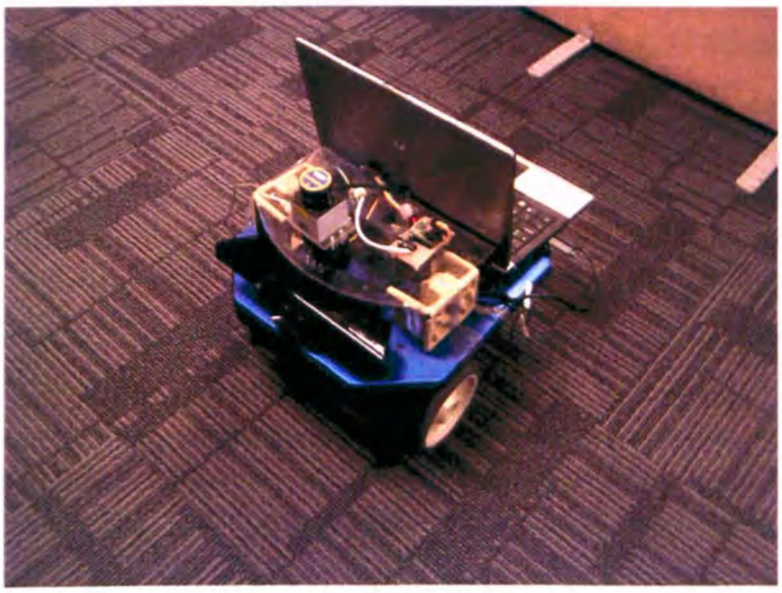


Figure 5.2: Robot example

lack of modularity implied that high levels of size and complexity made it difficult for robots to be constructed, evolved, and maintained [209]. Recent trends in robotics show a much more modular design and construction, in order to avoid such problems. Robots are usually equipped with numerous different types of sensors, to be used both for environment perception and data collection. Moreover, more robots are composed of heterogeneous interconnected hardware components capable of sharing information on a common protocol aiming at development time and cost reduction [208].

Following table 5.1, the hardware for each robot consists of an Erratic base robot, a Kinect camera, a pc webcam, a laser scanner and a laptop (see Figure 5.2). The Kinect sensor is front facing and is mainly used as an RGB input and depth measurement camera applied towards object detection and map building. The laser scanner, being more accurate than the Kinect (as will be further discussed in Section 5.1.3), is mainly used for map building and obstacle avoidance together with the wheel encoders available on the robot base. The PC webcam, attached to the laptop, is rear facing is was used to detect if a person is actually looking at the robot. Further information on sensors and hardware is available in Appendix Section A.2 and Appendix Section D.1 respectively.

5.1.2 Robotic Development Environment (RDE)

A framework which can handle various inputs coming from different sensors and capable of processing them in real time is necessary. These processes can be either SLAM, navigation, obstacle avoidance, human-machine interaction, object detection and identification or visual memory amongst others. Modular frameworks are expected to perform many different tasks in parallel, although not necessarily requiring all of the available functionalities and features together at the same time. These tasks must be coordinated and monitored properly to achieve the desired goal. If all components were to require different interfaces, system complexity would rapidly grow [209]. This therefore requires a modular or component-based middleware that provides customizable solutions based on the integration of the needed components to design and develop the required robotic system [210, 208]. Although modular design has many advantages in engineering, it raises some integration issues such as communication, interoperability, and configuration. All the different modules have to run in tandem without affecting one another and yet, when necessary they have to share information. Also, it is necessary for various software to access information coming from the same hardware. One particular example, encountered during the implementation, was the Kinect being used by both the pattern detector and also for navigation. This therefore necessitates the need of having a way to avoid conflicts when different pieces of software are accessing the same hardware. As further described in Appendix Section A.6.4 several robotic development environments (RDEs) exit. Out of all the available RDEs, ROS[1] (Robot Operating System) running over the Linux OS was chosen as the preferred middleware framework to be used for the robot configurations, for numerous reasons. The main reasons for using ROS, were that:

- the algorithms can run as separate multiple robot instances running as separate nodes / threads,
- the designed algorithms can run on both real robots and simulation with minimal changes,
- the input to the robot learning algorithm can come from any visual capturing device available on the robot / ROS network, and

[1]http://www.ros.org/wiki/

- robots can join or leave at any point. This also caters for the likely case that robots become inactive either due to lack of power or malfunction.

Further details on ROS are provided in Appendix Section A.6.4.

5.1.3 Implementation

This section looks at the actual implementation based on the platform shown in Figure 5.1 and the hardware architecture and software framework described earlier; thus allowing for real-time robot navigation, video image capture and efficient visual information sharing over the network. Visual memory sharing will be further discussed in Section 5.2.2.

As already discussed in Section 2.2.2 methods to create a full three-dimensional (3D) map of the environment do exist. Although possible to implement, it might not be a feasible option due to the power limitations of robots. It is argued that in this scenario, a full 3D map reconstruction was not a feasible option to have constantly running on the robot. This generally requires a dedicated machine with relatively large processing power, such as a graphics processing units (GPU), which was not available on the robot. Also, generally, 3D reconstruction still assumes a static environment [211]. An environment, especially one shared between humans and robots, is considered to be a highly dynamic one [212].

The robots can perform typical indoor mapping, localization and navigation functionalities based on either a laser range finder or Kinect. Kinect is a cheaper option to work with compared to laser range finder. Moreover the Kinect will also provide an RGB image without the need of an extra camera. However, during testing it was found out (and which was also somewhat expected), that having a laser range finder attached on the robot gives better map building and navigation capabilities. Also, the Kinect could not detect objects which are closer than 50cm to it and this affected the map building especially in cluttered environments. Again, the laser scanner performed better in such scenario. Kinect and laser specifications can be found in Appendix D.

For experimentation, map building[2] was done using a ROS implementation of GMapping[3] which uses Rao-Blackwellized particle filters [213]. This method consists of a highly efficient Rao-Blackwellized particle filer to learn grid maps from laser range data and is an effective means to solve the simultaneous localization and mapping (SLAM) problem [214]. In this approach each particle carries an individual map of the environment and an adaptive approach to reducing the number of particles is used. Figure 5.4 shows the map building using the method suggested in [214] by using using the Kinect sensor to measure the depth. The generated map, in grey, was superimposed

[2]http://www.youtube.com/watch?v=gK7U7tpExCg

[3]http://openslam.org/gmapping.html

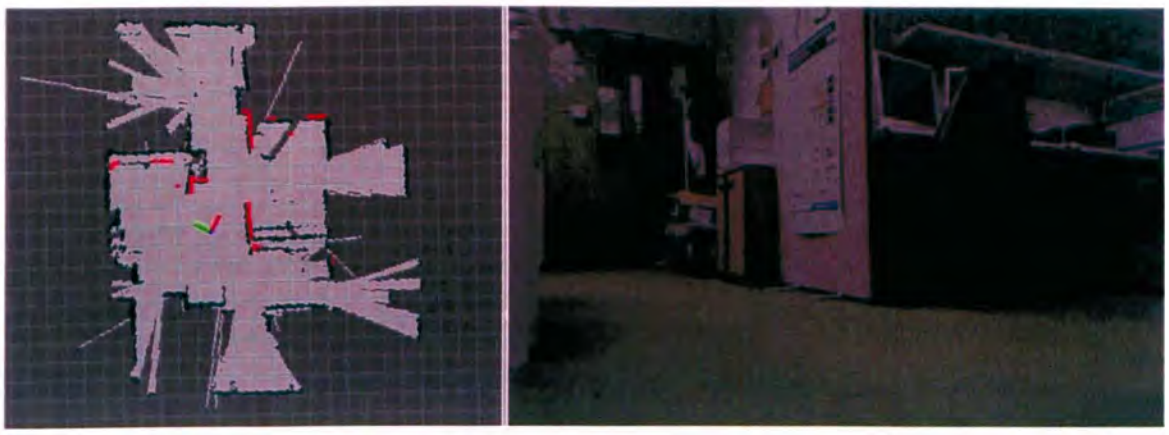


Figure 5.3: Robot SLAM, Navigation and video capture from Kinect

on the blueprint of the actual floor which is used as the ground truth. One can note that, although it gives a fairly accurate map building, there were some issues with the leftmost part of the map. Better results were obtained when this was repeated with the laser scanner as shown in Figure 5.5. Robot navigation[4] and obstacle avoidance were carried out using the ROS navigation stack. This implementation consists of an adaptive Monte Carlo-based localization, and costmap-based navigation and obstacle avoidance method as suggested in [215]. Further details on the robot setup using ROS can be found in Appendix Section D.2.

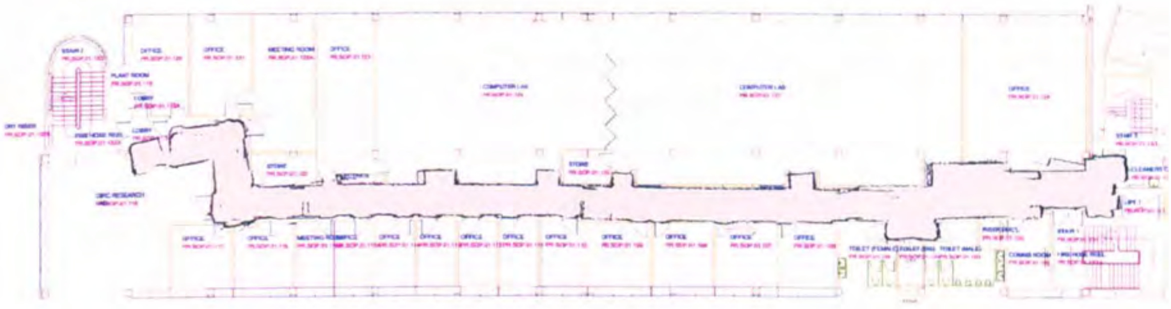


Figure 5.4: Map building using Kinect

5.2 Information fusion

Recent years have seen the first successful examples of augmenting the computational power of individual robots systems with the shared memory of multiple robots [28]. Although not directly related to this project, RoboEarth [28] is based on a similar principle centred around the idea of allowing robots to reuse and expand each other's knowledge by uploading it to a distributed database. As already highlighted in Section 2.5, shared world models include research on multiagent systems, such as RoboCup [110],

[4]http://www.youtube.com/watch?v=AbFz_02weco

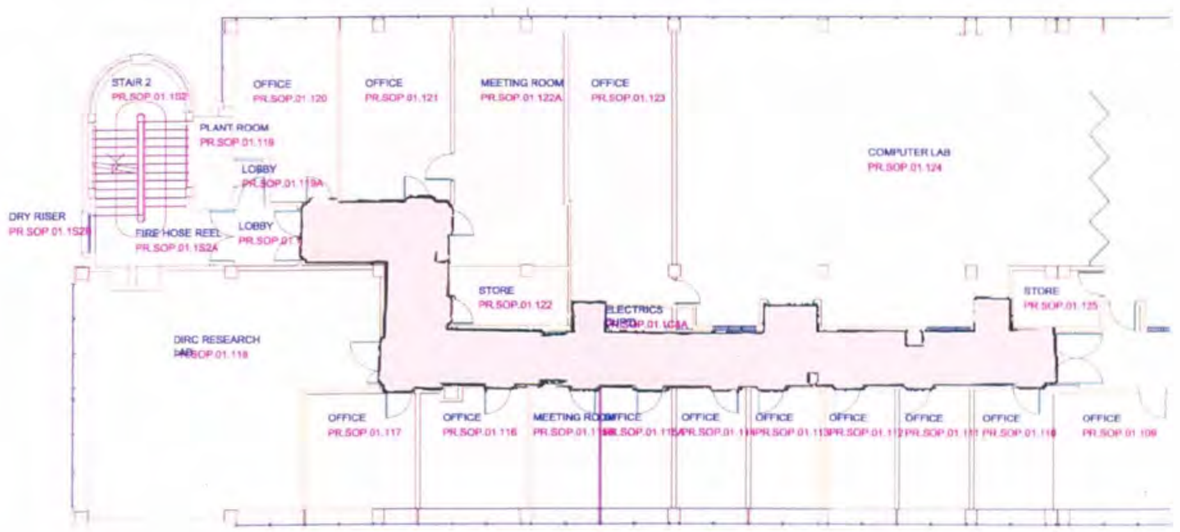


Figure 5.5: Map building using laser

where sharing sensor information has been shown to increase the success rate of tracking dynamic objects, collective mapping of autonmous vehicles, or distributed sensing using heterogeneous robots [28]. Information fusion refers to the acquiring, processing, and intelligently combining of information gathered by various knowledge sources and sensors to provide a better understanding of the phenomenon under consideration [216]. Multi-sensor information fusion involves the combination of information into a new set of information towards reducing uncertainty. This implies a process which generally employs both correlation and fusion processes to transform sensor measurements into updated states and covariances for entity tracking and which integrates various fields such as control theory, signal processing, bionics, artificial intelligence and mathematical statistics [29]. Several studies and comparisons on information fusion have been carried out [216, 4, 217, 218, 29, 19, 20, 16] with the main information fusion methods in robotic applications generally being: weighted average method, Kalman filtering, Bayes estimation, Dempster-Shafer evidential reasoning, fuzzy logic or neural networks [29].

Nakamura et al. [4] suggest that there are three types of information fusion: complementary, redundant and cooperative fusion as shown in Figure 5.6. Complementary information fusion is achieved when information is obtained using sensors or sources perceiving different properties of the environment. This is represented in Figure 5.6 with sensors $S1$ and $S2$ which provide different information, one coming from A and the other from B, merged together, thus achieving a broader information $(a + b)$ composed of non-redundant pieces a and b. Redundant information is when independent sources provide the same piece of information B, which is then fused to increase the associated confidence. Sources $S2$ and $S3$ in Figure 5.6 provide the same information, b, which is fused to obtain more accurate information, (b). Cooperative information is

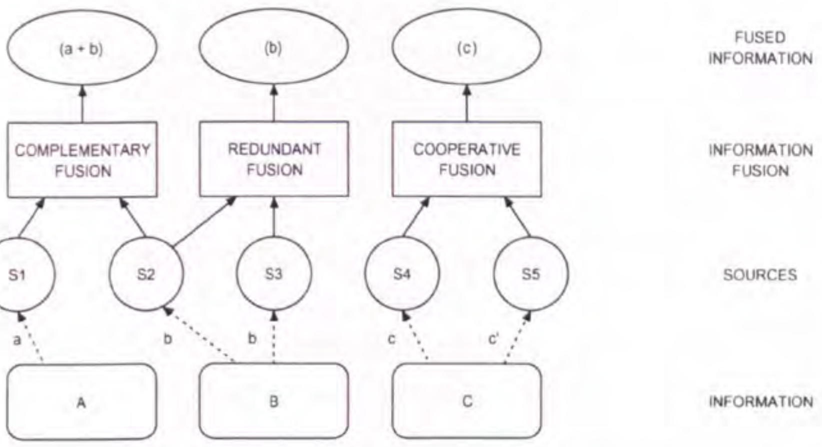


Figure 5.6: Types of information fusion based on the relationship among the sources [4]

when the information provided by independent sources is fused into new information (usually more complex than the original data) that, from the application perspective, better represents the reality. Sources $S4$ and $S5$, in Figure 5.6, provide different information, c and c', both coming from C that are fused into (c), which better describes the scene compared to c and c' individually. Two of these information fusion methods were considered, namely cooperative fusion in Section 5.2.1 and complementary fusion in Section 5.2.2. In both cases the following approach was considered:

- middleware framework approach as described in Section 5.1.3,
- distributed approach, allowing for complete decoupling between active nodes and services and where any active node (sensor or service) in the system may join a service that suits its purpose,
- the framework allows for pervasive (ubiquitous) computing within the human environment where all available services are published on the entire system, and
- communication between active nodes is peer-to-peer.

5.2.1 Cooperative fusion - stationary and mobile sensors

The multi-robot platform suggested in Section 5.1 was applied in [19] and [20] as part of the DHS project described in Section 1.2.1. By integrating computer vision and robotics components, it is shown that it is possible to monitor a large area using fewer sensors providing a scalable solution commensurate to the dimension of the monitored environment. The architecture is composed of stereo cameras and mobile robots, and the data captured by these heterogeneous sensors are combined in a decentralised manner. The combined information was obtained using a data fusion process that uses a reliability assessment of information sources.

Two distinct stereo camera systems were, using different vision approaches [19]. One stereo vision system was applied to reason on object manipulation events, whilst the other system was used to detect an event such as a person leaving a bag in the corridor. The results from either of these two systems were encapsulated in a string message and sent via a wireless network to the multi-robot system which, on alarm, dispatches a robot to monitor the region of interest. The ultimate goal was that of maximising the quality of information gathered from a given area, thus implementing a heterogenous mobile and reconfigurable multi-camera video surveillance system. In [19] the implementation of the multi-robot environmental monitoring used in this project was developed and tested both on two erratic robots and on many simulated robots. The team of robots monitors the environment while waiting for receiving event messages from the vision sub-system. A Bayesian filtering method was used for sensor data fusion. In particular, a particle filter was used for the sensor inputs and event detection layer. In this way, the probability density functions (PDFs) describing the belief of the system about the events to be detected are described as sets of samples, providing a good compromise between flexibility in the representation and computational effort. Localisation and mapping are based on a standard particle filter localisation method and a well-known implementation GMapping that has been successfully experimented as described earlier in Section 5.1.3.

This work was then extended to provide an enhanced multi-robot coordination and vision-based activity monitoring techniques [20]. In [20] a distributed, multi-robot solution to environment monitoring, in order to detect or prevent undesired events, such as intrusions, leaving unattended luggage and high temperatures (such as a fire) is provided. The problem of detecting and responding to threats through surveillance techniques is particularly well suited to a robotic solution comprising of a team of multiple robots. For large environments, the distributed nature of the multi-robot team provides robustness and increased performance of the surveillance system.

5.2.2 Complementary fusion - sharing visual memories

This work extends the ideas suggested in Chapter 4 to build visual memories of video streams to a multi-robot scenario. Several robots can be used to create a visual memory of the environments in a faster and more efficient manner. Each robot in the team is required to memorise its own area and at the same time share some of its acquired knowledge with its peers in an efficient and compact way. As it happens with human memory, details of a scene are retained and memorised images can be blurred or somewhat unclear [219]. However, sufficient information is retained to recall relevant information from memory about a part of the scene [174]. Likewise, the proposed algorithm does not produce a perfect photographic memory, but rather retains image representations, which contain meaningful information about the explored environment. In this method each node consists of an 80×60 pixel greyscale grid representing an evolving memory image. One basic way of having a global understanding of the environment is to have each robot surveying its own area, generating their own set of clusters and then feeding them into another learning network to create a common central memory. This however has some disadvantages. Aside from the fact there is reliance on a centralised system, robots would only know their area and would be totally unaware of what other robots are experiencing in other areas. This means that if a robot is faulty or without any battery power left, all the information obtained from the robot would be lost, unless it has already been provided to the central system. A distributed system to complement the above idea is required so that if one robot ceases to operate along the process, some of its most relevant information will be retained. It is therefore suggested to have several robots each with its own visual memory within a distributed environment. Robots are to memorise what they see and also accept incoming visual information from neighbouring robots, diagrammatically shown in Figure 5.7. Taking inspiration from the "island model" genetic algorithm [220] each

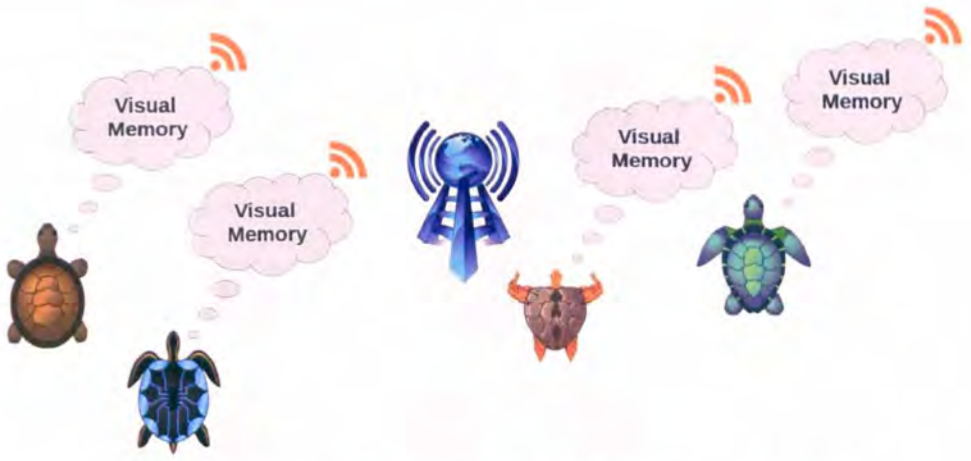


Figure 5.7: Concept behind the distributed visual memory

robot will start generating clusters of similar images within its own visual memory. This method revolves around the concept of migration where each island (in this case a robot) periodically exchanges a portion of its population (nodes) with other islands. Each robot starts generating clusters of similar images within its own visual memory. The average of each cluster is then calculated and a single image is produced thus generating one image per cluster (keyframe). One of these generated images is then selected at random and shared with the other robots. With the suggested method, one can have both a distributed and centralised system working together. The distributed system consists of robots which memorise mostly their environment with some influence from other robots and the centralised system to have a general understanding of all the environments being monitored. One way to do this in a distributed manner is to have a "visually impaired" robot together with the other "visually enabled" robots. This means that this robot would only receive inputs coming from other robots and share its own clusters based on these inputs. Referring to Figure 5.8 the noise cluster 5 is most likely generated by the "visually impaired" robot itself since in the first iteration it could only share noise as it had seen nothing before. At times this cluster can evolve into meaningful images but at others, as in this case, it might linger in the robot network. Each robot will operates using the procedure presented in Algorthm 3.

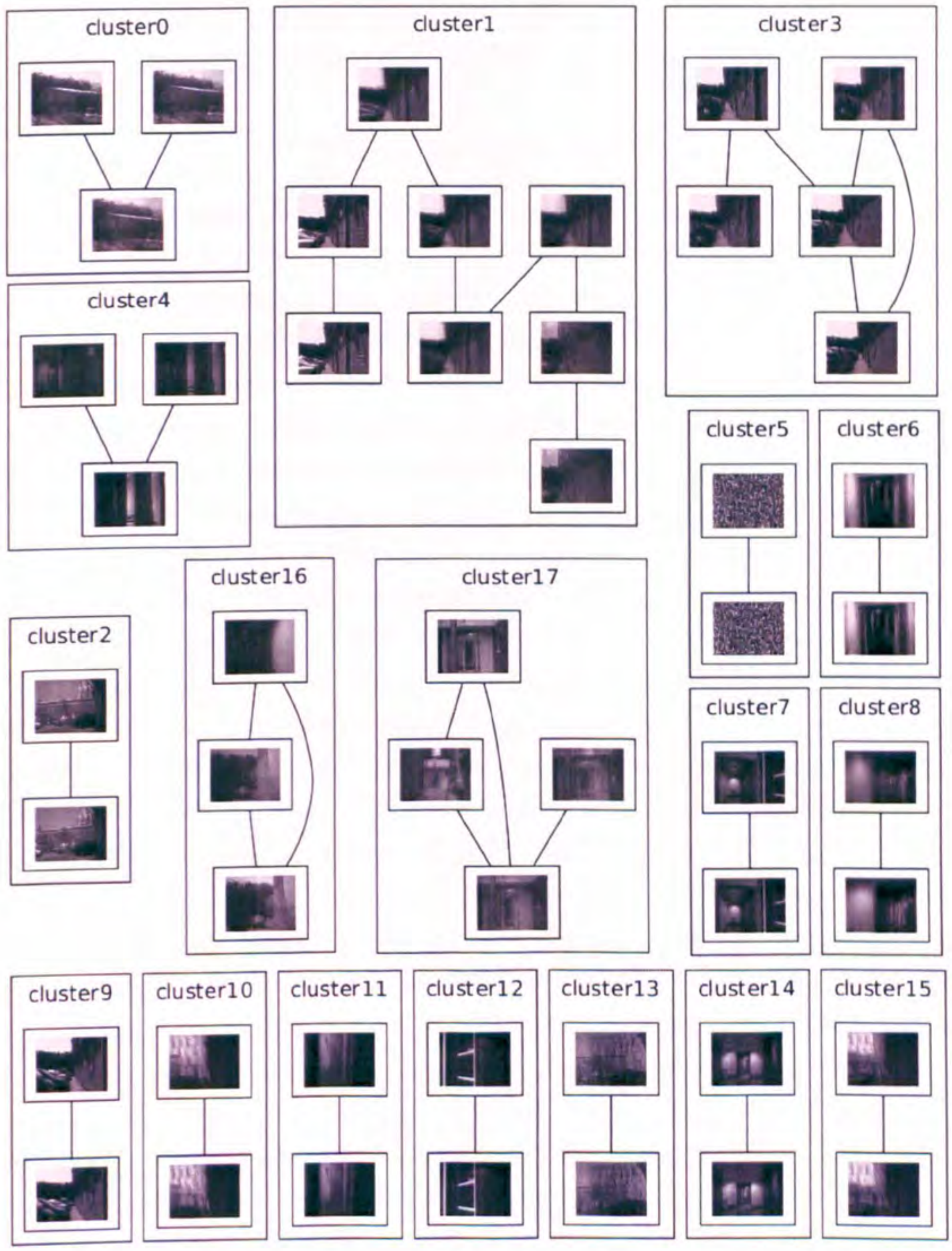


Figure 5.8: Visually Impaired Robot - This robots only memorises what other robots share

Algorithm 3 Robot Learning and Sharing Procedure

```
while (Stopping Criterion = false) do
    - Capture image from the environment
    if (broadcast image from other robots = true) then
        - Accept broadcast image
    end if
    - Scale and convert captured images to greyscale
    - Feed images into memory learning algorithm {learning algorithm starts generat-
    ing its own clusters}
    if (specified or random time elapsed = true) then
        - select a cluster at random and broadcast its average image (keyframe)
    end if
end while
```

In the proposed algorithm each robot is the "expert" of its area, however it will have enough information from the other robots, to know what else exists in the surroundings. Figure 5.9 shows a summary of how cluster sharing takes place between robots. Robots 1 and 2 generate an average cluster (keyframe) image which is shared with Robot 3. Robot 3 generates its own keyframe which in turn is shared with the other robots. One can note that the outcome fidelity is reduced. Similar to humans, when person A and person B say something to person C, it is highly unlikely that person C will relay accurate information to person D. Robot learning is similar to that of a child. Initially its knowledge will be "blurred". As time passes, its clusters will be better defined and therefore it will start sharing "better" information. Same applies to when it has to share its knowledge. A child will have unclear or "blurred" information which will become more clear by time [219]. The main advantage it that of initializing seeds within the peer robots with information about scenes which were not previously seen by that robot. This seed would allow robots to learn a new environment faster, if it happens to be the similar to one already visited and shared by other robots.

In this study, the way on how individual learning evolves with the change in number of robots and image injection rate from other robots was analysed. Only video inputs were required. The video streams were captured using a digital camera and fed into the ROS environment as a camera node to which each robot subscribed using its memory node. A range between two and five robots was used, each having a different input video sequence all of same length (2500 frames). Robots 1 and 2 surveyed two different corridors, Robot 3 was moving along an outside passageway between two buildings, Robot 4 was moving outside along a pavement in a car park and Robot 5 was in a road leading to the car park. The GNG parameters and the maximum number of nodes was kept fixed, so as to analyse the effect of varying the number of robots and the rate of sharing. The maximum number of nodes in the GNG was set to 50 and a new node was inserted every 5 iterations. In the proposed algorithm, the best matching

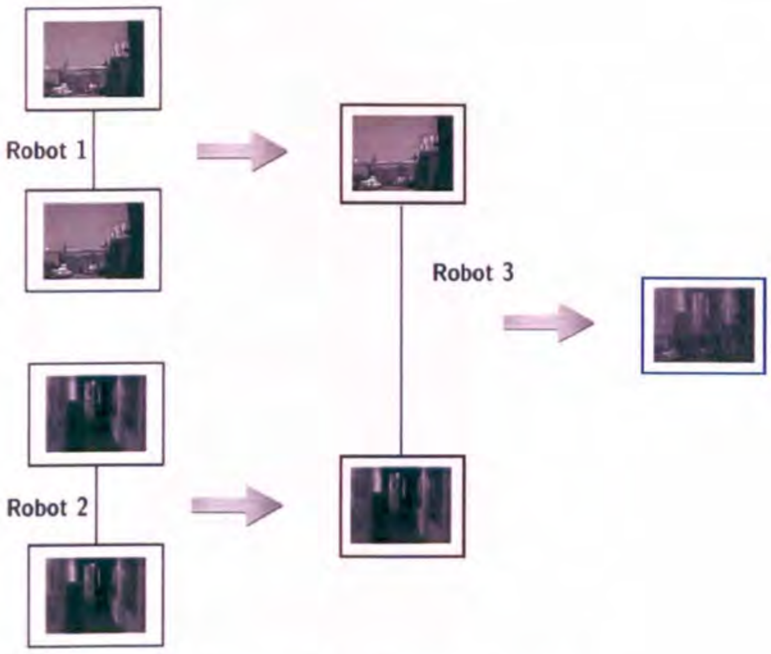


Figure 5.9: Cluster Sharing

unit coefficient is set to 0.95 and that of its neighbours to 0.001. The maximum edge age was set to 2 and for every iteration the error of each node is decreased by a factor of 0.005. For each set of robots (2, 3, 4, 5) a different sharing rate was used (one image shared by each robot every 5, 15, 25, 35, 45, 55 iterations). This led to a total of 84 different graphs. 12 for 2 robots, 18 for 3 robots, 24 for 4 robots and 30 for 5 robots. Each graph was checked for which nodes in the visual memory were not from the robot's input but rather from the common pool by manually comparing it to the ground truth data. The percentage sharing between robots was then calculated using

$$\frac{100}{n}\sum_{i=1}^{n}\frac{x(i)}{z(i)}$$

where $x(i)$ is the number of nodes within the visual memory of robot i not originating from the onboard camera and $z(i)$ is the total number of nodes within that network. The obtained values are shown in Table 5.2 and the outcome was plotted in Figure 5.10 showing the percentage of memory originating from the other robots (y-axis) versus frame sharing rate (x-axis). Table 5.3 shows the equation for each curve.

The higher the rate of sharing between the robots, the higher is the percentage of shared memory between the robots. A monotonic curve with negative gradient could therefore be assumed. Given a number of robots and a desired percentage of shared memory to be stored, the frame sharing rate should be set accordingly. In order to find the best fitting curve a 3 robot case scenario was used. The sharing rate was varied from 2 up to 40 in steps of 2. As shown in Figure 5.11 various curves where fitted and their R-squared value noted. Out of all the curves fitted, the best match was that

50 neurons per robot

Injecion rate from other Robots		Number of Of Robots				% memory originating from other robots			
		2	**3**	**4**	**5**				
	R1	5	2	5	10				
	R2	2	4	5	10				
5	R3		7	4	10				
	R4			4	6				
	R5				20				
		7	**13**	**18**	**56**	**7.00**	8.67	**9.00**	**22.40**
	R1	1	5	3	12				
	R2	3	2	2	4				
15	R3		7	0	4				
	R4			4	13				
	R5				8				
		4	**14**	**9**	**41**	4.00	**9.33**	4.50	16.40
	R1	3	5	1	7				
	R2	0	6	1	4				
25	R3		2	2	5				
	R4			0	8				
	R5				5				
		3	**13**	**4**	**29**	3.00	8.67	2.00	11.60
	R1	3	0	1	7				
	R2	0	2	1	4				
35	R3		3	1	7				
	R4			2	9				
	R5				5				
		3	**5**	**5**	**32**	3.00	3.33	2.50	12.80
	R1	1	5	0	7				
	R2	0	0	5	1				
45	R3		0	3	4				
	R4			5	5				
	R5				1				
		1	**5**	**13**	**18**	1.00	3.33	**6.50**	7.20
	R1	1	2	2	7				
	R2	2	1	0	4				
55	R3		0	3	3				
	R4			6	10				
	R5				3				
		3	**3**	**11**	**27**	0.03	0.02	0.06	0.11

Table 5.2: Multi robot sharing

Number of Robots	Best Fit Curve	R^2 Value
5	$y = -5.599\ln(x) + 31.176$	0.8825
4	$y = -1.472\ln(x) + 9.639$	0.2489
3	$y = -2.966\ln(x) + 15.197$	0.6263
2	$y = -2.029\ln(x) + 9.8927$	0.8241

Table 5.3: Curve Fitting

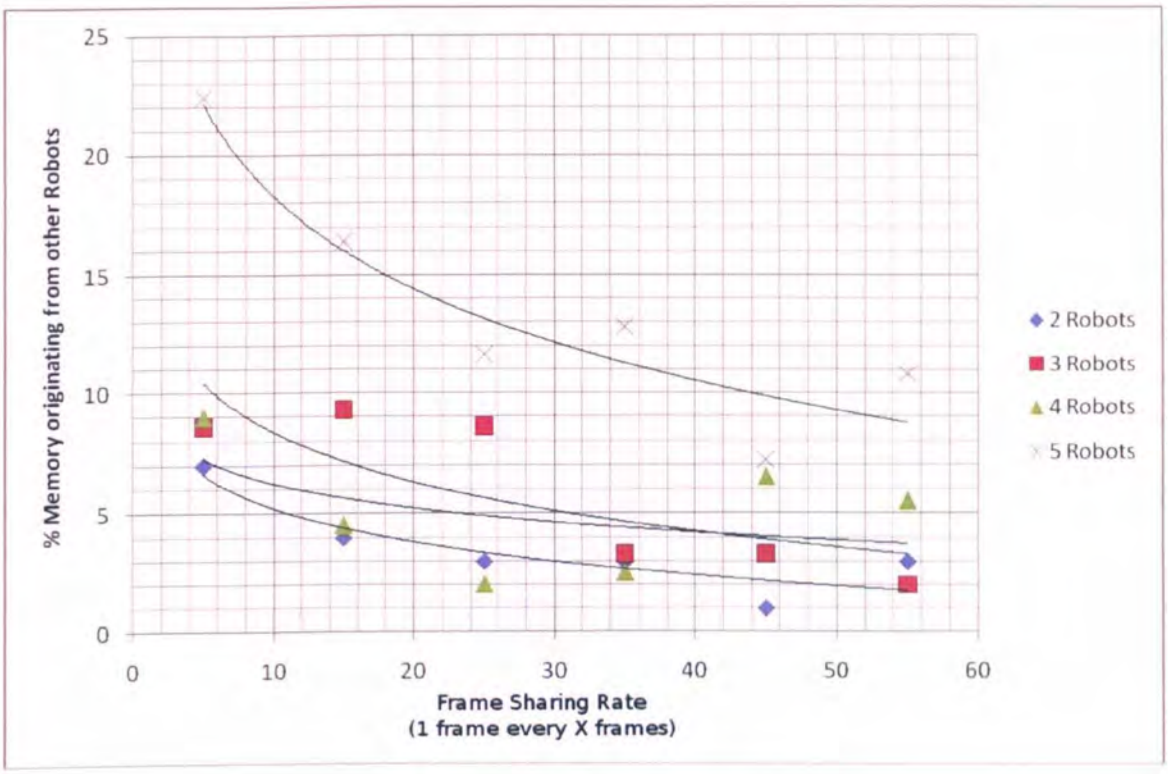
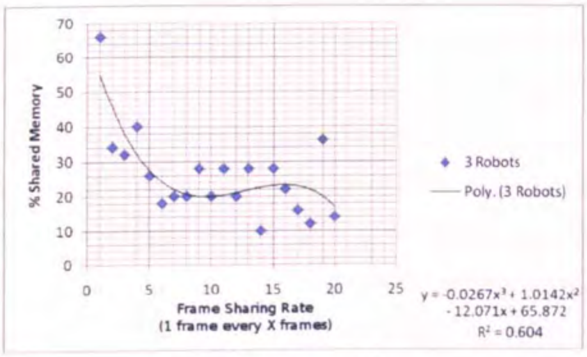


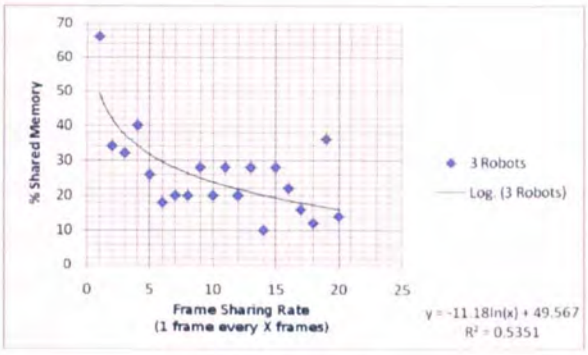
Figure 5.10: Percentage memory sharing (y-axis) vs frame sharing rate (x-axis) for various team sizes.

using a polynomial of order 3, followed by a log, then power, exponential and finally the least accurate being the straight line approximation. When varying the number of robots (from 2 up to 5), the best overall curve fitting performance was given by the log curve with an R-squared value of over 0.8 for the 2 and 5 robot case and a 0.6 for the 3 robot case. The 4 robot case performed at 0.2, still higher than the straight line approximation.

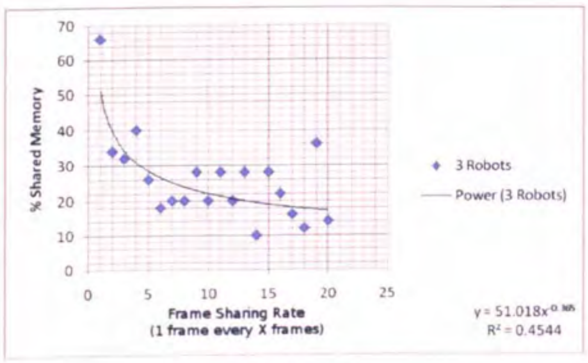
Due to the initialisation process of the GNG and the random nature of cluster image selection to feed other robots, the content of each visual memory will be different (i.e. not repeatable). This means that Figure 5.10 cannot be reproduced exactly for every run, however the general negative trend still holds. If the rate of sharing is low robots will tend to learn only their environment with low influence from the neighbouring robots. If, on the other hand, the sharing rate is too high, then, it might be that some robots could be overwhelmed with information coming from other robots and end up learning what others are memorising rather than building a memory of their environment. As the results tend to indicate, this situations becomes more acute as the number of robots increases.



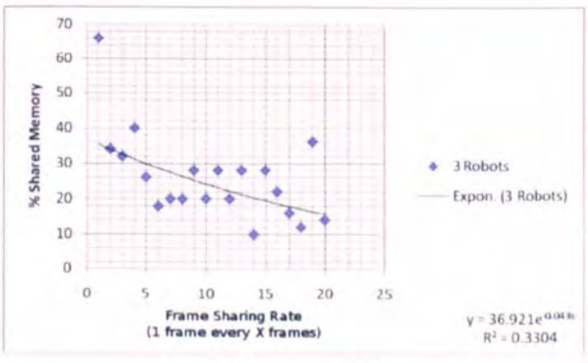
(a) Polynomial



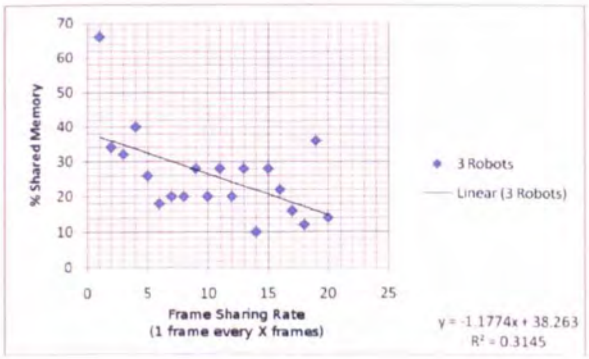
(b) Logarthmic



(c) Power



(d) Exponential



(e) Linear

Figure 5.11: Curve Fitting

5.3 Summary

This chapter looked at how the multi-robot platform introduced in Chapter 1 was implemented, looking at the best robot hardware architecture and software development framework for the application. In the implementation, the same hardware could be accessed by different nodes and processed information was made globally available. This chapter looked at how various sensors spread over the environment can be used, useful information is extracted, shared and fused for a better representation of the environment. Non overlapping sensors implies better sensor utilisation although leading to less (if no) redundancy. Different kinds of information fusion were discussed with particular interest on cooperative fusion and complementary fusion, where most of the work was focused. Cooperative fusion was used within the area monitoring scenario, by fusing information coming from various heterogeneous inputs. Complementary fusion was used for team visual learning by sharing visual memories.

"Any one whose disposition leads him to attach more weight to unexplained difficulties than to the explanation of a certain number of facts will certainly reject my theory."

Charles Darwin, *English Naturalist (1809 - 1882)*

Chapter highlights:
This chapter summarises the achievements in overcoming three main challenges, namely:

- autonomy,
- real-time operation, and
- fast deployment;

thus advancing the state of the art in real-time multi-robot perception and environment understanding.

6 Conclusions

The work presented in this thesis covers a method for the visual exploration and automatic interpretation of an uncharted environment by a team of robots, automatically learning new and interesting scenes within the environment. Whilst robots operate automatically, they cooperate by gathering, sharing and storing visual information in a compact and dynamic representation to reflect the visual input extracted from a changing environment. The state of the art in multi-robot vision was reviewed in Chapter 2 and various possible ways for robots to interpret incoming data from sensors or from other robots through the network to collectively achieve the expected targets highlighted in Chapter 1 were studied. In part based on biological inspiration, Chapter 3 focused on robot perception and how these techniques can be applied to make robots aware of their dynamic environment. Focusing mainly on robot vision, techniques to locate salient regions in a scene and the capability of robots to identify specific objects such as faces and tags were studied. The robots' performance was then enhanced in Chapter 4 by focusing on methods endowing robots with learning capabilities, allowing them to learn keyframes in real-time from a sequential input video stream and store them into a flexible topologically linked graphical representation network. Chapter 5 looked at the capability of robots sharing visual memories with other robots, possibly residing in different areas within the building. Moreover, methods introduced in earlier chapters are combined onto a complete platform so as to have robots capable of sensing, processing and learning their environment allowing for object detection and identification, SLAM, navigation and obstacle avoidance. Also, robots are capable to work in a heterogeneous environment, successfully sharing and fusing information between mobile robots, fixed cameras and humans for smart monitoring of complex public scenes.

6.1 Achievements

The work presented in this thesis looks at advancing the state of the art in intelligent monitoring applications by providing a simple, real-time yet meaningful representation of the environment by using a team of robots enhanced with learning mechanisms and computer vision techniques. Processes required to store visual information about the environment surrounding a robot in a compact representations and in re-usable format. Also, as in the case of AAL, the environment is shared by both humans and robots. This implied that when designing visual markings, as described in Section 3.2.1 and presented in [22], these had to be salient for both humans and robots alike. Following the work suggested in [24] results obtained from controlled experiments using human subjects were used to indicate on how visual information can be extracted and interpreted.

As highlighted in Section 1.4, three main areas, namely environment monitoring, handling of visual information and information sharing were tackled. Moreover, so as to achieve the aims, in this thesis three main challenges had to be overcome, namely autonomy, real-time operation and fast deployment. The achievements in overcoming these challenges are highlighted next.

6.1.1 Autonomy

As described in Chapter 5 and also presented in [19] and [20], a practical solution for environment monitoring using robots equipped with cameras was provided. These robots are also endowed with map building, localisation, navigation and obstacle avoidance capabilities to operate within a complex environment. These processes are essential for robots to be able to move freely, without bumping into team robots, surroundings or getting stuck in dangerous places. The team of robots is able to autonomously navigate and coordinate successfully with its peers.

6.1.2 Real-time operation

Real-time operation was an important challenge to be overcome for this application. When working in a dynamic environment, robots have to react to unforeseen circumstances such as the circumvention of obstacles. This is also the case with HRI where robots have to interact with humans. Without real-time capability, autonomy as suggested in Section 6.1.1 would be limited. Moreover, working with computer vision is considered as being very computationally intensive. This becomes even harder when considering the processing power and energy limitations of the robots. The methods suggested in Chapters 3 and 4, together with the work presented in [22], [23] and [25] took this into consideration. Saliency detection, as described in Chapter 3, still poses some challenges to successfully operate in real-time on a robot, mainly due to the numerous filters being applied. A compromise to overcome this challenge is to provide a compact encoding of saliency by combining saliency detection with visual memory techniques as suggested in [24]. A machine learning mechanism (the GNG mechanism as suggested in Chapter 4) allows the robots to memorise salient areas previously undetected by a robot within the environment. Once learnt, recalling memory is much faster than recalculating saliency, thus providing the robots with a real-time solution for a more selective searching capability.

6.1.3 Fast deployment

Another factor of great importance for the usefulness of a particular platform is the time required to have everything up and running and working in a robust manner. In Chapter 5, a multi-robot platform which, once designed to handle all the necessary tasks, could be deployed within a relatively short time (hours) was presented. Considering the heterogeneous environment, a multi-robot platform capable of hosting several different hardware on a common middleware framework was presented. This allows for a modular approach and sharing of resources without creating conflicts. Another factor is the ability to cater for scalability. This becomes more important as the number of robots increases. This modular approach also gives the great advantage of reusing available tools without the need to reconfigure the whole setup. In relation to fast deployment, the following items were successfully tackled:

- Map building can be done using SLAM techniques in a relatively short time and shared between devices thus making it available to mobile security guards and other robots working in the same area.
- With the proposed platform, new robots or hardware can be added, removed or relocated with minimal configuration changes.
- Taking into account the use of tags suggested for AAL in Chapter 3, there is no need to retrain the robots as the suggested method is robust and can operate under various environmental conditions. Moreover, these tags are easy to produce and affix to specific places for guidance or for the retrieval of metadata [22].
- With the visual memories suggested in Chapter 4 and extended to a multi-robot scenario in Chapter 5, robots can start collaborating to the learning as soon as they join the team [25].

6.2 Future work

Although good progress is being made, there are still numerous open problems which need to be tackled for successful and efficient integration of robot security systems in human environments. One of the challenges addressed in this thesis is to provide a flexible graphical representation of the robot's environment. When it comes to visual memories, research can be further extended onto methods using better feature extraction techniques thus providing a more faithful and descriptive representation the environment. Such features could possibly include colour histograms and edges, to mention some. Visual memories can even be extended to learn more specific objects. This would require further research into efficient feature extractors and descriptors which would allow the objects, not necessarily the same, but falling under the same category, to be identified from various angles and orientations, automatically cropped and fed into memory; this further leading into researching methods for automatic semantic descriptions of the memorised objects.

Optimisation of saliency algorithms is another open problem which can be tackled to allow for real-time operation. These algorithms are still computationally more intensive compared to their well trained pattern detector counterpart. Having more computationally efficient saliency detectors would allow for a better top-down and bottom-up combination; this again with the intention of embedding them onto robot platforms to extract salient objects within the environment automatically be used as an input feed the robot learning mechanism.

Apart from platform robustness, robots, especially those used in surveillance, have to operate in complex environments where crowded situations may occur at random times. Such environments are generally cluttered and hard to train for. This thesis mainly looked at how robots can be made aware and react to their environment. The capacity to successfully track people and luggage is also another open problem which is being tackled. Taking again the airport baggage reclaim as an example, the shape, size and pose of people carrying luggage varies considerably. Methods to automatically detect the density and entropy of crowds need to be developed. These can then be used to further enhance security and surveillance by being able to send personnel or even automated robots to disperse or aggregate crowds accordingly using the appropriate means. Most of the crowd analysis mechanisms are based on fixed cameras using background subtraction for person tracking. These would fail in the case of using mobile cameras such as the ones attached to a mobile robot. This is another unsolved challenge in the area of robot visual intelligent monitoring.

A
Additional Literature

A.1 Robot Types

As researchers, industrialists and engineers keep investing time, money and resources towards robotics research, many varieties of robots have been created, making it possible for us humans to conquer even the most remote location on the earth, being sky high, buried deep in the ground in mines, the deepest ocean abyss and even outer space. Figure A.1 illustrates some of the most common types of robots. Robots could have a general design such as a wheeled robot or a more complex humanoid robot, to more tailor specific designs. Other designs also include flying and diving robots. Multiple legged robots can also be found. These are generally capable of navigating on uneven terrain such as that found in woods, parks. Tree climbing robots such as Woody, Treebot and Rise [221] can also be found within in the research community. Some of the most commonly used robot types used will be discussed hereunder.

A.1.1 UGV - unmanned ground vehicles

Unmanned ground vehicles (UGVs) are vehicles that operate in contact with the ground and without an onboard human presence. UGVs can be used for many applications where it may be inconvenient, dangerous, or impossible to have a human operator present. These can generally be classified as either holonomic or nonholonomic constraints. A holonomic kinematic constraint can be expressed as an explicit function of

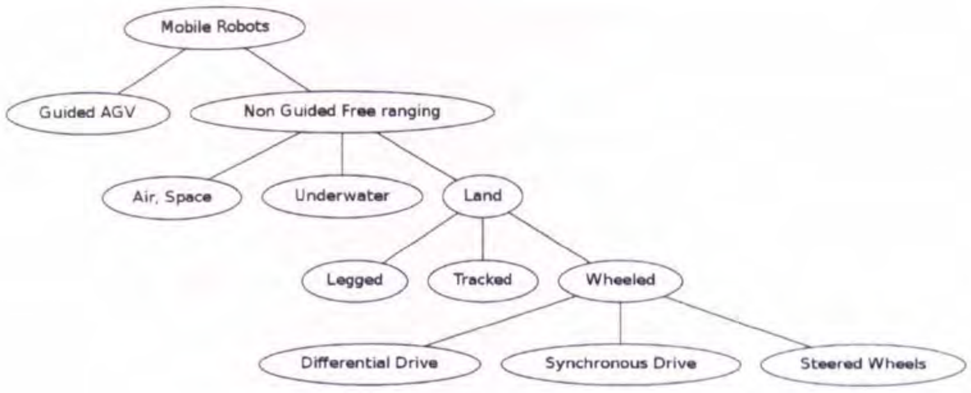


Figure A.1: Main mobile robot types

position variables only and does not involve derivatives of these variables. A nonholonomic kinematic constraint requires a differential relationship, such as the derivative of a position variable. Furthermore, it cannot be integrated to provide a constraint in terms of the position variables only. An example of a nonholonomic constraint is the movement of a wheel on an axle. The velocity of the contact point between wheel and ground in the direction of the axle is constrained to zero; the motion of the wheel is therefore subject to a nonholonomic constraint [222]. Although there are more configurations for land moving robots (car like, walking robots, etc.), differentially driven (nonholonomic) [223] and three omnidirectional wheel platforms (holonomic) [224] will be considered for our study as they provide a good platform for our experiments. An Erratic Base[1] and Rovio robot[2] were used for our experiments. Differential drive is a method of controlling a robot with only two motorized wheels. The term 'differential' means that robot turning speed is determined by the speed difference between both wheels, each on either side of the robot. The differential steering system is familiar from ordinary life because it is the arrangement used in a wheelchair. If both drive wheels turn in tandem, the robot moves in a straight line. If one wheel turns faster than the other, the robot follows a curved trajectory about a point a distance R away from the centre of the robot, changing both the robot's position and orientation. If the wheels turn at equal speeds, but in opposite directions, the robot pivots about its geometric centre. Omnidirectional driving can be obtained by using three omnidirectional motorised wheels placed at 120^o to each other. With this configuration the robot can be moved in any direction independent of the orientation. This comes handy when the robot is fitted with a fixed front facing camera. The robot can move sideways and still use the camera to scan the frontal view. This is not possible with a differential drive robot as there is no way the robot can move sideways.

[1]http://www.videre.com (possibly discontinued)
[2]http://www.wowwee.com/en/support/rovio/

A.1.2 UAV - unmanned aerial vehicles

Having robots capable of flying autonomously and simultaneously whilst perceiving and reacting to their environment pose even further challenges to the robotics community. The current challenge is to have UAVs that can fly autonomously or semi-autonomously and carry out a purposeful mission. At present, the majority of UAV systems in the literature are not autonomous and require a team of pilots, engineers and technical personnel to carry out a mission. Common tasks of UAVs include: search and rescue operations, remote areas inspection and sensing, hazardous material recovery, real-time forest fire monitoring, surveillance of sensitive areas (borders, ports, oil pipelines), etc. [225]. Flying robots provide additional constraints when compared to their simulated and even real ground counterparts mainly due to their construction and data capture mechanisms. Some such challenges include limited payload capability, very limited battery life, the need of a more robust stability control mechanism [225] and a faster reaction time to external forces such as wind. The sensors used on these vehicles play an important factor. Using GPS by itself is not always reliable for UAVs due to effects like shadowing or multipath in city-like environments [44]. Ar.Drone 2.0[3] is a relatively successful 'toy' robot which could easily be controlled from mobile devices and which is equipped with a frontal and a base camera. This makes it ideal for UAV robotics research within an indoor environment. The application domains of 'search and rescue' and 'intelligent monitoring' with UAVs, and in particular UAVs endowed with machine vision technology have been partially explored in [226, 227, 228, 229, 230, 231]. Due to the power, weight and computation limitations inherent in the field of UAVs, the sensor-feeds and the algorithms that process them usually yield slow and noisy update rates including delays, and in the case of monocular vision, unknown scale of position measurements [232]. This poses new challenges to the research community. Until present, most video analytics have been carried out on video streams acquired from stationary surveillance cameras or partially mobile cameras, such as PTZ or cameras mounted on robotic rovers with limited scope. Liu and Dai [233] present a survey of computer vision applied in aerial robotic vehicles highlighting visual navigation, aerial surveillance and airborne visual SLAM. Choi et al. [228] propose an algorithm based mostly on optical flow and geo-location for target tracking from a UAV with no knowledge assumed about the target, apart from being in motion. Riehl et al. [229] present a receding-horizon cooperative search algorithm that jointly optimizes routes and sensor orientations for a team of autonomous agents searching for a mobile target in a closed and bounded region. They show that this algorithm makes effective use of agents and have also successfully tested this algorithm a board of two small UAVs equipped with cameras [229]. In [234] the relative pose

[3]http://ardrone2.parrot.com/

of two robots in absolute scale and in real-time using one monocular camera on each robot is calculated. This is achieved by combining measurements from the onboard inertial sensors on each platform with information obtained from feature correspondences between cameras using an extended Kalman filter (EKF). This forms a flexible stereo rig, providing the ability to treat the two robots as one single dynamic sensor, which can adapt to the environment and thus improve environmental mapping, obstacle avoidance and navigation. Goodrich et al. [226] emphasize on the practical aspects of visual-based aerial search, present, analyze a generalized contour search algorithm, and relate this search to existing coverage searches. Goodrich et al. [226] also showed that constructing temporally localized image mosaics is more useful than stabilizing video imagery. Andriluka et al. [227] address the task of automatically finding people lying on the ground in images taken from the on-board camera of a UAV. They evaluate various state-of-the-art visual people detection methods in the context of vision based victim detection from an UAV. Andriluka et al. [227] discuss their strengths and weaknesses and demonstrate that by combining multiple models the reliability of the system can be increased. The top performing approaches in this comparison are those that rely on flexible part-based representations and discriminatively trained part detectors. Andriluka et al. [227] also demonstrate that the detection performance can be substantially improved by integrating the height and pitch information provided by on-board sensors thus providing a substantial step towards making autonomous victim detection for UAVs practical.

A.1.3 UUV / AUV - unmanned / autonomous underwater robots

In [235], several application areas for underwater robots are highlighted. As also suggested in [236] underwater robots can be used in case of oil spills, to measure the waters temperature, salinity and also the movement of the dispersed oil in order to detect just how polluted the water was. Underwater crawler robots can be used for geological and biological surveys.

A.1.4 Robots in outer space

Outer space is a harsh environment with extreme temperatures, vacuum, radiation, gravity, and great distances, human access is very difficult and hazardous and is therefore limited. Yoshida [237] explores the latest achievements of robots used in outer space highlighting technologies used, current research and technological challenges ahead. Of latest interest in space robotics is the Curiosity Rover successfully sent to Mars in 2012[4].

A.2 Sensors

For a robot to be able to react with the environment it should be able to sense its surroundings. Various sensors are available. Some of these include wheel encoders, inertial sensors, cameras and ultrasonic, infra red and laser range finders (Figure A.2). Traditionally robots have used sonar, infra-red or laser range finders to measure distance from obstacles and identify areas which are safe to navigate in. Lasers are more accurate in providing depth information however they are comparatively expensive. Sonars are good for nearby obstacle detection however one would have to deal with interference issues. Infra red sensors are more accurate that ultrasonic but suffer similar problems. The introduction of the Kinect in November 2010 provided researchers with a sensor which is relatively cheap compared to laser and previously available RGBD cameras. These are able to provide both a digital image and depth information by integrating a camera for images and an infra red structured matrix transmitter and receiver for depth. Some of these sensors including the Kinect are described in more detail in Section D.1. Borenstein et al. [238] provide an introduction to various 'classical' sensors available for robot localisation and sensing of the environment, however, since it is relatively old, not much focus is put on vision. Also it does not include the current advances in digital imaging when it comes to depth estimation and 3D positioning. Digital cameras have become very cheap and easily available in recent years. They have opened up new possibilities as a sensor for robot perception. Cameras have the advantage of gathering colour information; something which is not available with other sensors. Whereas vision provides rich data, useful information might not be readily available; however this can be extracted using the appropriate mathematics. There are various formats in which this information can be stored the most common being RGB, LUV and HSI. These, so called colour spaces use three components to reflect colour information and facilitate the specification of colours in some standard, generally in an accepted way [239]. Pascale [240] provides a review of the various colour spaces available whilst Bhattacharyya [48] surveys some colour image preprocessing and segmentation techniques.

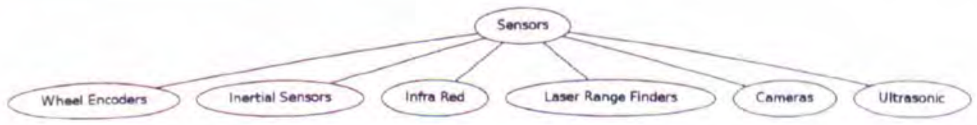


Figure A.2: Sensors

[4]http://en.wikipedia.org/wiki/Curiosity_rover

For wheeled mobile robots, the conventional and simplest way of determining the robots position is by counting the number of pulses from the wheel encoders and thus calculate the relative position from the initial starting point. This is more commonly referred to as odometry. With the successful introduction of vision in robot applications, visual odometry became another option. Visual odometry algorithms [241, 242, 73, 243, 244], are capable of reconstructing the scene and, at the same time, estimate the position of the camera with respect to an external coordinate frame. Within the computer vision community this is better known as egomotion and is defined as the 3D motion of a camera within an environment. The process of estimating a camera's motion within an environment involves the use of visual odometry techniques on a sequence of images captured by the moving camera. This is typically done using feature detection to construct an optical flow from two image frames in a sequence generated from either single cameras or stereo cameras. Using stereo image pairs for each frame helps reduce error and provides additional depth and scale information [245]. Nister et al. [242] present a system that estimates the motion of a stereo head or a single moving camera based on video input. The system operates in real-time with low delay and the motion estimates are used for navigational purposes. Armangue et al. [246] survey several methods for 3D motion estimation unifying the mathematics convention which are then adapted to the common case of a mobile robot moving on a plane. Table 1 in [246] highlights some of the main motion recovery methods.

A.3 Image comparison techniques

Several image comparison methods exist. Comparison techniques could either make use of image pixel values directly, such as the Image Euclidean Distance(IMED) or else features or other metrics could be first extracted and then the comparison is carried out depending on the similarity of features. Tuytelaars and Mikolajczyk [247] provide a very good and detailed survey on feature extraction and image comparison techniques. In the coming sections some methods which were investigated and used for our applications are discussed.

A.3.1 Image Euclidean distance

The Euclidean distance between between two points is the length of the line segment connecting them and is given by the Pythagorean formula. This method can be used to quantify by how much two images differ by calculating the colour distance per pixel. This method works well when the images are fairly static, however it might give false results if there is a minor camera shift since all the respective pixels would contain considerably different values even though both images would be similar overall.

As a way to overcome this problem, Wang et al. [190] proposed an image distance measurement which takes into consideration both the spatial relationship and the gray level relationship between pixels.

A.3.2 Joint probability distribution

Another possible solution for image comparison is that of calculating the entropy of a Joint Probability Distribution between two images of a scene. This generally involves the creation of a probability density function from a normalised histogram which is then matched using some PDF matching techniques such as the Kullback-Leibler divergence (See B.3 for more details) or the Jeffrey distance to mention some.

A.3.3 Histograms

Histograms provide a compact representation of an image, thus requiring less memory space. Image histograms are invariant to rotation of the image around the optical axis, making it particularly attractive to omnidirectional cameras. Also, histograms are not very sensitive to small translations [248]. This method uses only elementary image processing, is potentially fast compared to other approaches, and is generally straightforward to implement. The normalized cross-correlation often yields good results for comparing images on a global level. One simple method is to build feature histograms for each image, and choose the image with the histogram closest to the input image's histogram. Histogram matching generally works well for matching images which are very similar to the input image and is generally immune to small changes within the image, however it generally fails with rescaled, rotated, or discoloured images. Histograms are created by first deciding on the number of histogram bins and then set each bin to a fraction of the whole range. Assuming two 8-bit greyscale images, a bin for all the possible greyscale values (0 to 255) is created. For each pixel, get the pixel value and increment the respective bin. When all pixels have been checked, the histogram is normalised thus creating a Probability density function (PDF). Various histograms can be created for each image (e.g. one for Red component, another for Green and another for Blue). Generally, just comparing image colours won't be enough and other features (of which histograms can be created) are used. Other histograms could be created by counting corners in image areas obtained using methods described next. These could all be compared and the image with the least error between the two compared images could be considered as the most similar.

A.3.4 Feature extraction and keypoint matching

In computer vision, and more specifically in object recognition, many techniques are based on the detection of points of interests on objects or surfaces. This is done through the extraction of features [68]. Tuytelaars and Mikolajczyk [247] argue that the term detector has been used to refer to the tool that extracts features from the image, e.g., a corner, blob or edge detector. However, this only makes sense if it is a priori clear what the corners, blobs or edges in the image are, so one can speak of "false detections" or "missed detections". If this is not the case, the term 'extractor' would probably be semantically more correct. However, they also argue that the term 'detector' is widely used and is therefore probably wiser to keep the same terminology throughout. Local features are the standard representation for wide baseline matching and object recognition [247]. Also, if one needs to track these points from a moving camera, possibly on a robot, typically for navigation, localisation or map building, a reliable feature has to be invariant to image location, scale and rotation. Given a set of features, matching is performed by associating keyframe pairs between a couple of frames [68]. There are two aspects concerning a feature; the detection of a keypoint, which identifies an area of interest, and its descriptor, which characterizes its region. Typically, the detector identifies a region containing a strong variation of intensity such as an edge or a corner, and its center is designed as a keypoint [68]. The descriptor consists of a multidimensional feature vector which identifies the given keypoint. Tuytelaars and Mikolajczyk [247] document the evolution of feature detection and examine some of the most widely used detectors, with a qualitative evaluation of their respective strengths and weaknesses.

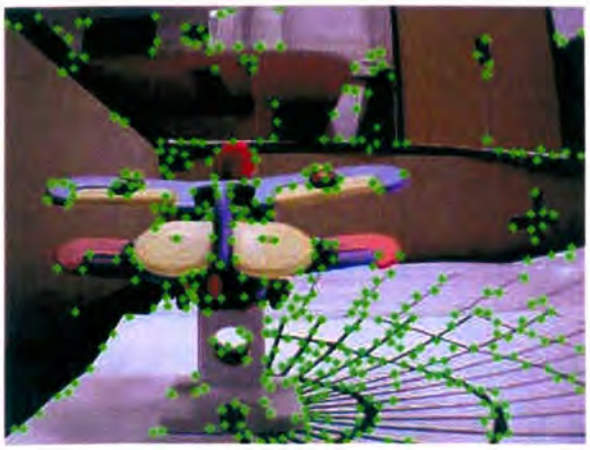


Figure A.3: Good features to track

When it comes to robotics, real-time tracking is essential, so a balance needs to be found between, extracting good features, describing them in an efficient manner and eventually track the features in different images / video frames. One possible solution to quantifying the similarity of two images is by using a corner / feature detection and matching algorithm. The more matched corners / features there are between similar images the more probable is that the two images reflect the same scene. Different detectors and descriptors have been used for mapping and localization using monocular or stereo vision such as the Harris corner detector [249], Shi-Tomasi [250] , Harris-Laplace [251], Scale invariant feature transforms (SIFT) [252, 253], Speeded-Up Robust Features (SURF) [254] or a combination of attention regions with Harris corners [60]. Certain parts of an image have more information than others (particularly edges and corners). The idea behind keypoint matching is that of finding these points and be able to compare them between images. One starts by finding good corners / features in each image. One possible way of doing this is by using Shi-Tomasi corner detector [250]. Figure A.3 shows the outcome for such detector. The green dots represent point features (corners) which would be good for tracking between images. These corners are then used in a Pyramid Lucas-Kanade optical flow to obtain the matched corners between the two images [255]. The Lucas-Kanade method is a two-frame differential method for optical flow estimation where the number of features correctly matched between the two images is used as a similarity measure. The Shi-Tomasi corner detector [250] is based entirely on the Harris corner detector [249]. In the Harris corner detector a score is calculated for each pixel using two eigenvalues. If the score is above a certain threshold, the pixel is marked as a corner. Shi and Tomasi suggested that the function to provide a score could be done away with and only the eigenvalues could be used directly to check if the pixel was a corner or not, thus making this detector perform better. SIFT (Scale-invariant feature transform) keypoints are arguably the most popular where SIFT descriptors are used to compare local features and they are often used for correspondence analysis and object recognition since they can be used to match images under different scales, rotations, and lighting. SIFT and SURF(Speeded Up Robust Features) combine a detection and description method. In particular, SIFT features were developed for image feature generation, and used initially in object recognition applications. SIFT features combine a method to extract stable locations in images and a description that enables to identify those points [60]. Coloured SIFT (CSIFT) [256] can be considered as an extension to SIFT which was mainly designed for gray images. CSIFT is more robust than its counterpart with respect to colour and photometrical variations. SURF was inspired by SIFT but several times faster. Juan et al. [257] described and compared SIFT, PCA-SIFT and SURF using K-Nearest Neighbour (KNN) to find matches and Random Sample and Consensus(RANSAC) to reject inconsistent matches from inliers. Principle Component Analysis (PCA) is a standard technique for dimensionality reduction. They confirmed that SIFT is slow and not good at illumination changes, while it is invariant to rotation, scale changes and affine transformations. SURF is fast and has good performance as the same as SIFT, but it is not stable to rotation and illumination changes. Gil et al. also made a comparative evaluation of interest point detectors and local descriptors in [64]. They considered the following interest point detectors, namely Harris Corner Detector, Harris-Laplace, SUSAN, SIFT, SURF, MSER and Kadir. For local descriptors, they considered SIFT, GLOH, SURF, Gray level patch, Orientation Histograms and Zernike Moments. They concluded that SURF and GLOH are the best descriptors and outperform SIFT. Hogman [68] also arrived to the same conclusion that SURF is much faster than SIFT. Moreover, Ahmed et al. [189], provide an optimised implementation of SURF based recognition technique aimed to run on small embedded robot platforms with limited processing resources. They claim a time reduction from 39 seconds to 780 milliseconds.

A.4 Learning mechanisms

This section extends from Section 2.4 looking in more detail at the various learning mechanisms available.

A.4.1 Supervised learning

Supervised Learning is a type of machine learning in which labelled training data is available and where a model of the data has to be created (learned). The algorithm is presented with an input set and an output set. The aim is to find a generalised function that approximates the relationship between input and output. This learned model will then be used to predict outcomes from the test data and hence classify the test data. When the output is in the form of a class label, it is termed as *classificaiton*. On the otherhand, *regression* is when outcome is a real number [46]. Reinforcement learning can also be used for supervised learning. In such scenario, the system would receive a delayed signal in the form of a reward or punishment from which the system would try to infer a policy for future runs [46]. The objective of reinforcement learning is to learn how to act in a dynamic environment from experience by maximising some payoff functions or minimizing some cost functions equivalently. In reinforcement learning, the state dynamics and reinforcement function are at least partially unknown. Thus the learning occurs iteratively and is performed only through trial-and-error methods and reinforcement signals, based on the experience of interactions between the agent and its environment [131]. Q-learning is a value learning version of reinforcement learning that learns utility values (Q values) of state and action pairs [131].

Traditional supervised techniques would generally involve either Generative methods or Discriminative methods [46]. Discriminative algorithms yield predictions by returning the probability of the label given the data ($P(L|D)$) whereas generative algorithms give the distribution of the data given the label ($P(D|L)$) thus possibly conditionally synthesizing new data or providing a more powerful representation of the data. Some of the Generative methods include Naive Bayes classifiers, Artificial Neural networks and principal component analysis (PCA) followed by classification. Some of the Discriminative methods include Support vector machines and Linear Discriminant Analysis. A combination of Generative and Discriminative algorithms can also be used as suggested in [258]. Bradski and Kaehler [46] suggest that if one has enough time to train the classifier, which however must run quickly, then neural networks, normal Bayes classifiers and support vector machines are a good choice. If moreover, one requires high accuracy then they suggest that boosting and random trees could be a better option. On the otherhand, if one only requires an easy test for good feature selection then decision trees or nearest neighbours might do the job.

There are certain problems however related to supervised learning. Numerous amounts of good labelled data needs to be available for proper classifier training. Labelling this data is boring. Moreover at times experts in the field might be required for proper labelling, such as in medical imaging. These experts might not be available. Also, the number of topic categories for classification might not be readily available or might change, possibly increase, as more data is provided. In such a situation unsupervised learning needs to be used as will be described next.

A.4.2 Unsupervised learning

Another kind of machine learning technique is unsupervised learning and is commonly used in clustering and dimensionality reduction. In this case no prior training is given and the algorithm iteratively learns to identify similarities in input data and classify them accordingly. In this kind of learning technique the algorithm is provided only with unlabelled data. Moreover, no feedback is provided from the environment. The aim of the algorithm is to find patterns in the input data which otherwise might be observed as unstructured noise.

Based on whether the problem to solve is classification, prediction or dimensionality reduction, and on the background knowledge of the space sampled various methods could be used. Some commonly used unsupervised learning techniques for dimensionality reduction include PCA [259], pLSA [260] and ICA [261]. Density estimation can be used for prediction by estimating some underlying PDF. Clustering is a fundamental data analysis method. It is widely used for pattern recognition, feature extraction, vector quantization (VQ), image segmentation, function approximation and data mining.

Clustering methods are generally based on either statistical model identification or competitive learning. K-means clustering and mixture models can be used to classify unlabelled real valued data and k-modes clustering for unlabelled categorical data given that the number of desired clusters is known a priori. Dirichlet Process Mixture Models (DPMM) algorithm tries to fit the best clustering model over the data can be used for clustering when the number of possible clusters is not known a priori [262]. Du [178] provides a review on clustering methods based on competitive learning. As already discussed in Section 4.2. SOM [177] is one of the most successful and widely used clustering network however with a fixed topology. GNG [181] on the otherhand, is a network which grows over time with evolving topology and with the ability of forming separate clusters.

Whereas most of the conventional classifiers are designed to operate in batch mode and do not change their structure online, incremental classifiers work on a per-sample basis and only require the feature of that sample plus a small amount of aggregated information. The Evolving fuzzy-rule-based classifier suggested by Angelov et al. [263] is another incremental classifier. This type of classifier is especially useful for the realization of incremental online and evolving schemes, further discussed in Section A.4.4. The authors state that such method has the important property of being able to start learning from scratch. Evolving systems are systems which experience a gradual change and learn from experience. The ability to evolve is a desired feature in robotics and is of particular interest in the field of visual memoires amongst others.

At times unsupervised Learning might provide some unexpected results. Generated clusters might not adhere with real world clustering. Also, there are times when it is hard to define if the clustering was done correctly or not as most of the real world problems are often subjective. One such situation is in segmentation. When prior information on the data to be fed the learning network is available, semi-supervised learning can be used. In this case some data can be labelled to guide the network. Also, some user suggestions giving feedback on the progress of the network can be incorporated.

A.4.3 Semi-supervised learning

As the name suggests, semi-supervised learning is in between supervised and unsupervised learning techniques. Semi-supervised learning involves function estimation based on both labelled and unlabelled data with the goal of reducing the amount of supervision required compared to supervised learning. This approach is motivated by the fact that labelled data is often costly to generate, whereas unlabelled data is generally not. The challenge in such techniques mostly revolves on how to handle mixed input data in such a way to perform clustering successfully up to the expectation of the user.

A.4.4 Online learning

As with other learning algorithm methods the main problem to solve with online learning is the task of making decisions with limited information [264]. Online learning consists of an algorithm which is used to predict labels as close as possible to the true labels for instances received sequentially [265]. The goal of the algorithm is to minimize some performance criteria. This model proceeds in a sequence of trials where the algorithm starts by receiving an instance. The algorithm would then predict the label for that instance and which is then compared with the later received true label of the instance. For example, the instances could describe the current conditions of the stock market, and an online algorithm predicts tomorrow's value of a particular stock. For this prediction the algorithm may attempt to minimize sum of the square distances between the predicted and true value of a stock. Another popular performance criterion is to minimize the number of mistakes when dealing with classification problems. Two popular on-line algorithms perceptron [265] and winnow [266] can perform well when a hyperplane exists that splits the data into two categories. These algorithms can even be modified to do provably well even if the hyperplane is allowed to infrequently change during the online learning trials [264].

Online learning requests for continual label feedback. Therefore, for any problem that consists of predicting the future, an online learning algorithm just needs to wait for the label to become available. The key defining characteristic of online learning is that soon after the prediction is made, the true label of the instance is discovered. This information can then be used to refine the prediction hypothesis used by the algorithm. Because online learning algorithms continually receive label feedback, the algorithms are able to adapt and learn in difficult situations. Many online algorithms can give strong guarantees on performance even when the instances are not generated by a distribution. As long as a reasonably good classifier exists, the online algorithm will learn to predict correct labels.

A.5 Robot visual memory techniques

This section looks at several techniques found in literature used to store visual experiences, mainly for robot localisation and navigation. In [91], [267] and [268] an omnidirectional camera is used with the advantage of a wide field of view, thus minimising the number of images required [267]. In [267] a hierarchical localization method for omnidirectional images based on the Fourier signature is proposed. In [91] the visual memory consists of a database of sorted omnidirectional reference images, including some topological information as well as some metric information. In [91, 241] it is also claimed that their hierarchical method reaches accurate metric localization with a minimal reference data set. Another solution, especially for the case when a standard perspective camera is used, can be that of extracting some features from the images that reduce the amount of required memory while retaining an unambiguous description of the image. Research in [86] integrates an image retrieval system with Monte-Carlo localization. The image retrieval process is based on features that are invariant with respect to image translations, rotations, and limited scale. Other techniques highlighted in [267] include a method to extract a set of Eigen images from the set of reference images and to project the images into Eigen spaces. Jones at al. [51] describe the use of appearance based methods to design simple visual processes for navigation and use cameras and odometry to sense their environment. They use zero mean energy normalised cross correlation between the observed image and the template image. Ulrich and Nourbakhsh [248] propose a system which is able to carry out robot localization both in indoor and outdoor environments by using colour vision. They take a topological approach where maps are represented by adjacency graphs. Nodes represent locations while edges represent the adjacency relationships between locations. Their localisation method is based on nearest-neighbour learning and operates in the histogram space. Cummins and Newman [269] state that most localisation and loop closure methods are based on similarity measures and image-matching techniques. In [269] however, the authors are not interested in the similarity between observations, but rather on the probability that the observations come from the same place. They adopt a bag-of-words representation in a probabilistic framework. Scenes are represented as a collection of attributes (words) chosen from as set (vocabulary). Mendes et al. [65] present an approach where paths are learnt by storing sequences of images and image information in a sparse distributed memory (SDM). Connections between paths are detected by exploring similarities in the images, using the same SDM, thus creating a topological representation of the paths. The robot is then able to plan paths and switch from one path to another at the connection points. Kit et al. [173] demonstrate a method to learn the distribution of visual features in an environment via a self-organising map. The model encodes spatially-distributed colour histograms of real world visual scenes captured by a camera moved through an environment. The colour distribution is learned as a histogram using a self-organising map with location (when available) and colour data. Colour histograms are insensitive to small view changes. Once trained, the network may be queried to provide a predicted histogram that can then be compared against the current true image histogram to detect changes. Kit et al. [173] state that the concept behind their representation is that the spatial data structure should be view-invariant but still retain enough fidelity for change detection. Their model consists of first acquiring image and spatial data (if available), calculating colour histograms from the images, and finally training a self-organising network on the histogram and spatial data (if available). Further information on colour histogram matching is provided in Appendix Section B.1.

A.6 Robot information sharing

This section extends over Section 2.5 and looks at robot communication. Networking, communication and collaboration methods requirements for successful information sharing.

A.6.1 Robot communication

Robots within the environment need to communicate, coordinate and share data and information between themselves with systems evolving into teams of robots. The main motivation for connecting robots together is for them to communicate in order to achieve a common mission in either a distributed or parallel manner [118]. Rocha et al. [42] suggest that communication may appear in three different forms of interaction namely by using (i) stigmergy (via the environment), (ii) implicit communication (via sensing) or (iii) explicit communication (via communication). As discussed in Section 2.1.2, stigmergy is a mechanism of indirect communication between agents or actions by using the environment itself. Communication between robots can be either implicit or explicit. Implicit communication is a mode of communication in which the robots sense the action of other robots through the latter's action on the same target. A robot would knowingly use its sensing capabilities to observe and perceive the actions of its team mates. Explicit communication, on the other hand, is a mode of communication in which the robots share information with each other by exchanging messages via a wireless communications network [95]. Put in other words, implicit communication occurs as a side effect of other actions whereas explicit communication is a specific art designed solely to convey information to other robots on the team [41].

Robots are becoming more autonomous and as a result there is the opportunity for such autonomous intelligent robots to collaborate in order to carry out more complex tasks. As will be discussed in Section A.6.3 they need to determine what jobs have to be done and who is best suited to perform them, according to some work objectives. As will be seen in Section 2.1.1, interaction with human observers or controllers may be also a factor to be integrated as seamlessly as possible [114]. In these cases, wireless communication provides the low cost solutions for mobile robot networks to cooperate efficiently. Wireless communication is vastly growing and hardware is becoming more readily available. Wang et al. [270] suggest that ad-hoc networking is the best option for mobile robots due to the fact that in most applications robots are most likely equipped with only low power wireless transceivers whose range is too short to allow direct communication with the data collection point, but sufficient to allow robots to communicate with close neighbours.

In a single robot system, most if not all of the 'reasoning' is generally done on the robot computer itself whenever possible, to allow for maximum autonomy. Robot networking was generally used only for simple data communication such as such as informing the robot of new position coordinates. Traditionally, robot communication architectures has been based on a client-server model between a robot and a server respectively inter-connected through an access point [118]. When it comes to multiple robots in a MRS however, the story is different. Apart from each robot working independently on its sensing and movement processing, more heavy data might need to be shared between robots, agents and other control systems in order to collectively decide the best operation, either through a centralised system or distributed system. These agents need to communicate and share data and information between themselves with systems evolving into teams of robots supported by wireless ad-hoc networks [118]. In an ideal situation robots should only transfer data which is necessary thus reducing network traffic and also the little data sent contains enough information to make decisions. Bandwidth is a precious resource especially if a robot's task involves transmitting huge data. Efficiency becomes increasingly important when scaling to a larger number of agents [270]. Sheng et al. [271] address the problem of how to reduce the data exchange among multiple robots when they carry out cooperative area exploration or coverage. Mosteo and Montano [114] address the task allocation problem from a point of view of networked robotic teams, its objective being to optimize the utilization of the resources available. This kind of research can be found under several labels like mobile ad-hoc networks (MANET) or networked robotic systems (NRS). It may also be seen as a specialized branch of cooperative robotics where the communication is explicit via the wireless medium and where it shares many elements of research with the field of wireless sensor networks (WSN). MANET is a collection of mobile nodes which form a temporary network using ad-hoc routing. The network often has a dynamic and unpredictable topology. NRS is a network consisting of mobile robots and static sensors which are connected over a wireless network. WSN is a medium to large mesh of networks typically using low-power nodes [114]. Networking mechanisms are further covered in Section A.6.2.

A true multi-agent problem necessitates restrictions on communication [85]. At any rate, while full, unrestricted communication can orthogonalize the learning problem into a basic single-agent problem, such an approach requires very fast communication of large amounts of information. Real-time applications instead place considerable restrictions on communication, in terms of both throughput and latency.

A.6.2 Networking

As mentioned earlier, distributed applications such as surveillance, and urban search and rescue (USAR) operations, require solid networking protocols so as to coordinate their activities with reliable communication. Ubiquity of communication between robots, sensors, monitoring centres and human security guards equipped with smart devices is essential for proper coordination in such applications. Moreover, in such systems, a working network infrastructure is generally not available or cannot be assumed to be reliable, so robots and other devices deployed should be able to communicate through ad-hoc networks [270].

As already mentioned in Section A.6.1 MANETs can be described as a fully distributed, autonomous and cooperative communication networks that can be effectively set up and operated without the need for pre-established infrastructures. In such a network there is no centralised administration or standard support services [270]. MANET is a collection of mobile nodes which form a temporary network. The network often has a dynamic and unpredictable topology. MANETs are self-organised wireless mobile networks that consists of mobile entities that can each assume the role of a data source, destination or router. In such networks, data can be directly transmitted from source to destination if both interconnected by an enabling technology such as Zigbee (IEEE 802.15.4), Bluetooth (IEEE 802.15.1),Wifi (IEEE 802.11).

Several studies on cooperative distributed networking and cooperative functionalities have been carried out [272, 273, 274, 275, 276]. Witkowski et al. [275] provide a dynamic triangular robot ad-hoc infrastructure for multi-robot systems in disaster scenarios. Sugiyama et al. [276] suggest an ad-hoc chain network connecting a base station and rescue robots so as to explore distant spaces in disaster areas. Such scenarios require time-critical information to be routed in a multi-hop manner to the appropriate IP-based entities. Additionally, such network entities are autonomous in nature and have limited battery capacities. Thus, the cost of such communications is often considered to be in terms of the energy spent for overheads due to routing [277]. Some of the metrics to analyse a good robot network system include energy consumption, connectivity, throughput, accuracy, security, robustness and bandwidth efficiency [270].

A.6.3 Robot cooperation and coordination

Robots are said to cooperatate when there is a situation in which several robots operate together to perform some global task that either cannot be achieved by a single robot, or whose execution can be improved by using more than one robot, thus obtaining higher performances [97]. Cao et al.[93] define cooperative behaviour as a subclass of collective behaviour that is characterized by cooperation. They state that given some task specified by a designer, a multiple-robot system displays cooperative behaviour if, due to some underlying mechanism, there is an increase in the total utility of the system.

Cooperative multi-agent systems are ones in which several agents attempt, through their interaction, to jointly solve tasks or to maximise utility. Due to the interactions among the agents, multi-agent complexity can rise rapidly with the number of agents or their behavioural sophistication [85]. Cooperation between robots provides three potential advantages, namely efficiency, reliability and robustness, and specialisation [42]. By using multiple robots, sensors are more spatially distributed then if a single robot was used, thus providing the possibility of capturing data at a faster rate with less uncertainty about the environment. This also introduces redundancy and therefore, the failure of any particular robot does not necessarily compromise the overall mission success. Also, robots with different sensory or motion skills may have complementary and specialized features that overcome their individual limitations and increase the system's total utility.

Particularly challenging domains for multi-robot domains are those tasks that are inherently cooperative. These tasks cannot be decomposed into independent subtasks to be solved by a distributed robot team. The utility of the action of one robot is dependent upon the current actions of the other team members.The success of the team throughout its execution is measured by the combined actions of the robot team, rather than by individual robot actions [41]. Cooperation among unaware robotic agents is the weakest form of cooperation [97]. A fundamental part of a MRS is that the robots should be aware of each other's existence. Awareness alone however does not implicitly specify that the robots will collaborate and cooperate. This implies that one of the first pre-requisites for effective collaboration relies on the possibility for the robots to communicate properly with each other as described in Section A.6.1.

Intentional cooperation is clearly not a prerequisite for a MRS to exhibit coordinated behaviour, as demonstrated by minimalist or emergent approaches [98]. In such systems, individuals coordinate their actions through their interactions with each other and with the environment, but without explicit negotiation or allocation of tasks. When compared with emergent cooperation, which is a result of acting upon selfish interests, intentional cooperation is usually better suited to the kinds of real-world tasks that humans might want robots to do. If the robots are deliberately collaborating and cooperating with each other, then, they must share information on a common protocol [97]. Intuitively it can also be expected that if need be, humans could deliberately also cooperate with these robots [98]. In [97], Iocchi et al. present an analysis of MRS by looking at their cooperative aspects. They propose a taxonomy of MRS and a characterization of reactive and social deliberative behaviours of the MRS as a whole.

Good coordination methodology should always consider incorporating past knowledge to accelerate initial decisions driven by an expert system or offline learned prediction models, be able to adapt knowledge based on real-time behaviour and should not solely rely on the use of communication and intelligent perception but look at alternative techniques to complement and replace these technologies [110].

Coordination is a form of cooperation where the actions performed by each robot take into account the actions executed by the other robots within the same team resulting in a coherent and high performance operation [97]. Coordination is not a prerogative of the cooperative MRS, in fact there exist robotic systems in which coordination between the members is required, but the robots have different goals which often are not related to each other. For example, the industrial robots often share tools and coordination is needed to avoid interferences, which take place when a single indivisible resource is requested by multiple robots [97]. Sentis et al. [278] address the problem of coordinating great numbers of vehicles in large geographical areas under network connective constraints. A large exploration task was simulated where it was demonstrated that the required constraints can be effectively enforced while optimizing the exploration goals. Sentis et al.[278] suggested advanced skills in MRS by using hierarchical potential fields, grouping together various field objectives to accomplish the performance requirements in response to high-level commands. Their framework calculates trajectories that comply with priority constraints while optimizing the desired task objectives in their null spaces. A model-based dynamics approach was used, providing a direct map from field objectives to vehicle accelerations, yielding smooth and accurate trajectory generation.

Distinction has to be made when commands are issued to robots working individually at a microscopic level as opposed to a team of robots at a macroscopic level [95]. Distinction also needs to be made between the roles that a robot must perform in a team and the targets on to which the roles are performed. Roles are associated with tasks and targets are associated with the objects to which the tasks are directed. Single target scenarios tend to provide more scope for implicit communication and more macroscopic control, whereas multiple target scenarios tend to provide more scope for explicit communication and more microscopic control [95]. In contrast, single role scenarios tend to proved more scope for implicit communication and more macroscopic control whereas multiple role scenarios tend to provide more scope for explicit communication and more microscopic control.

A.6.4 Robot software development frameworks

As the capability of robotic application increases, extensive infrastructure support together with expanding development of support for autonomic computing would be required [207]. Since the original concept of robot development environments (RDEs) came about, their main aim was that of providing robot and component developers, coordination developers, and system architects with practical solutions for a consistent and simple interface for design, programming, debugging and coordination of robots that can then be scaled up to consumer solutions [279, 209]. Such frameworks could be also used for the execution and maintenance of robotic architectures as part of application deployment [207]. Frameworks should provide a simple and clean component interface to invoke specific functionality, monitor task progress, and update the goals of running tasks [209] with the ultimate development platform being the one which would allow the general population to program robots with ease [279]. Mohamed et al. [208] highlighted several challenges which need to be overcome by robot development frameworks, namely:

- simplifying the development process,
- support communications and interoperability,
- providing efficient utilization of available resources,
- providing heterogeneity abstractions,
- supporting integration with other systems,
- offering often-needed robot services,
- providing automatic recourse discovery and configuration, and
- supporting embedded components and low-resource-devices.

Biggs and MacDonald [279] reviewed various robot programming frameworks, making a distinction between manual and automatic programming frameworks. Manual frameworks require the user/programmer to create the robot program directly, by hand. These can be either text-based or graphical. Automatic frameworks on the other hand, generate a robot program as a result of interaction between the robot and the human: there are a variety of methods including learning, programming by demonstration and instructive systems. Due to the variety of methods available, one of the main issues during the planning stage, is that of deciding which is the best framework to use. Kramer and Scheutz [207] conclude that common features will be increasingly expected in all frameworks, strengthened by the interoperability of certain mechanisms. The creation of a set of (possibly de facto) standards, would lead to an increasing number of predefined components that can be expected in any given RDE [207]. To date however, although some platforms are more preferred than others within the robotics community, no de facto standard has yet been reached [280]. Several open source, freely available RDEs for mobile robots exist. TeamBots, ARIA, Player/Stage, Pyro, CARMEN, MissionLab, ADE, Miro, MARIE, Orca, UPnP Robot Middleware, ASEBA, The PEIS Kernel, ORiN, RSCA, The Middleware of AWARE, Sensory Data Processing Middleware, Distributed Humanoid Robots Middleware and WURDE are just some of the RDEs and middleware that have been extensively analysed and compared [207, 208]. More recent developments saw the creation of ROS (Robot Operating System) [281, 282] which will be described in more detail next.

ROS (Robot Operating System)

ROS is a robot-specific middle-layer solution for distributed computation and message passing. It allows easy integration of sensor drivers and data processing components including both off-the-shelf and in-house components. Each piece of hardware or software runs on its own node and publishes over a topic. These topics are available for all the other nodes. When a particular node wants to read data coming from a particular sensor it will subscribe to the topic on which the sensor is publishing and the data together with its time-stamp are available. This allows for multiple sensors and multiple users to access the data without conflicts. The distributed nature of ROS allows each independent component to function with some degree of independence and facilitates extensibility [282]. ROS provides the biggest advantage of having several developers able to share their developed software even if implemented on other robot platforms. Due to the nature of ROS, if data is published correctly on a topic, its source is irrelevant, thus making the system very versatile and highly scalable. ROS also gives the possibility of having one ROS master running on one machine and other machines / robots connected to it as slaves over wireless Ethernet. Solutions for having several ROS masters working in a distributed fashion and sharing topics over Ethernet are available. Although still in its infant stage ROS is becoming very popular and the number of users is exponentially increasing [283]. ROS is in a unique position as a middleware as it enjoys high community participation, a regular release interval, good simulation environments and, the capability to reach from the smallest devices up to the high powered service robots [280].

However, ROS is still not perfect and some problems with the use of ROS were highlighted in [280]. These arise mainly due to the fact that ROS requires every master node to hold the complete namespace in its memory. Together with the flat design of the multimaster implementation, every change to the namespace system must be propagated through the whole system, to be available to every node. Schneider [280] suggests that a hierarchical system, where the master nodes form a tree structure which only forwards the parts of the namespace might be more appropriate. Another problem of ROS is the verbosity and complexity of its XML-RPC protocol which lays a heavy burden on nodes in wireless sensor networks. However, considering all the pros and cons, ROS was still considered as the best development environment for the robots.

B
Additional Algorithms

B.1 Scene comparison by colour histogram matching

As already seen in Section A.2 various colour spaces exist. In this section LUV and more specifically the U and V channels are used for basic scene understanding and to compare scenes based on their chroma levels. Images are matched by comparing their chrominance levels. The U and V components from the RGB values of the first image are obtained. This is followed by creating a probability density function (PDF) of U vs V by a binning and normalisation process of values with the outcome being similar to Figure B.1. This is repeated for the second image.

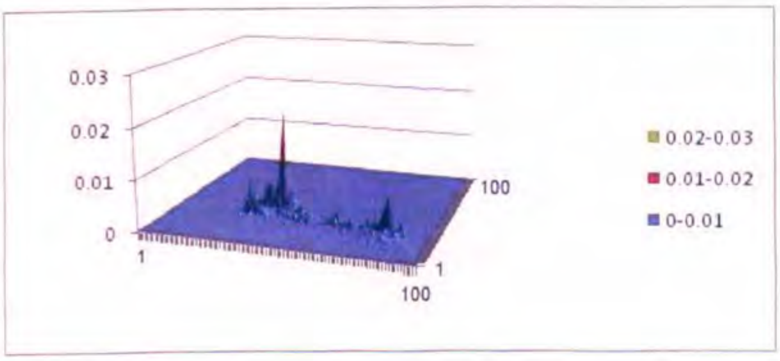


Figure B.1: U vs V PDF

The two PDFs are then compared using Kullback-Leibler divergence. By using this method the similarity of two scenes based on colour can be defined. The more similar the colour content is the smaller the divergence and thus the smaller the value. Moreover, if one assumes that the object of interest is always in the centre of the image a Gaussian weighting based on Equation (B.1) could be included in the binning part, with the value of the bin being increased by a value dependent on how close to the centre the pixel is. The further away the pixel is from the centre the lower its contribution will be. The Gaussian curve is graphically depicted in Figure B.2. Figure B.1 shows a normalised PDF of an image.

$$\frac{1}{2\pi\sigma^2}\exp\left(-\frac{(x_0-x)^2+(y_0-y)^2}{2\sigma^2}\right) \tag{B.1}$$

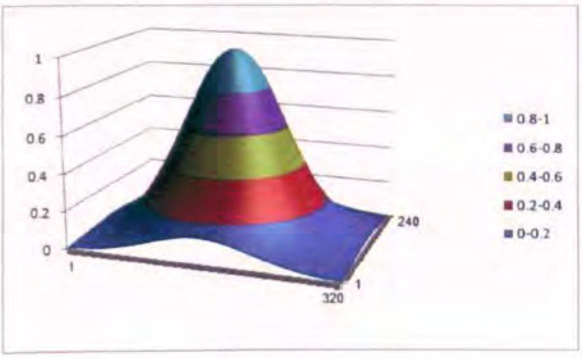


Figure B.2: Gaussian Curve

During experimentation, various images of size (320 × 240) were captured from an SRV-1 surveyor[1] robot. Figure B.3 and Figure B.4 show the sample sets of images used for scene comparison. Image Set 1 in Figure B.3 consists of Images of a toy plane with bright colours. Images were taken at intervals of approximately 10^o by the mobile robots. A similar set, this time without the toy plane and just the background scene, as seen in Figure B.4 is used. Figure B.5 and Figure B.6 show the outcome of the PDFs produced from 2 particular images. The PDF shown in Figure B.5 c) was generated from image Figure B.5 a) and the PDF shown in Figure B.5 d) was generated from image Figure B.5 b). Similarly for Figure B.6. In Figure B.6 two images from the same image set containing the toy plane were used. Although all the PDFs vary for each image, as can be visually noted, the difference in the PDFs of Figure B.3 appears to be more than that in Figure B.4. This intuition is confirmed when using the Kullback-Leibler Divergence. For Figure B.5 the divergence is of 10.54 whereas for Figure B.6 is much smaller and amounts to 0.886. The outcome from the match in Figure B.6 gives more confidence to the robot that the scene in the two images is more similar to the case in Figure B.5. When comparing any two images from the same set the Kullback-Leibler divergence is smaller to when comparing images by crossing the sets.

[1]http://www.surveyor.com/SRV_info.html

The difference in chroma between the images thus indicates that there is a different coloured object in the scene.

For experimentation, a colourful toy plane was used as the object of interest. The reason for this is that such colours are not normally found in the natural environment, thus making the difference between the object and background more apparent. Also, in an ideal situation, with lighting fixed and static background (which was not in this case), the change in the U vs V plot peaks will be proportional to the volume of each colour within the image. From the results shown hereunder, the outcome gives a good indication and gives a basic understanding of what is happening within the field of view of the robot.

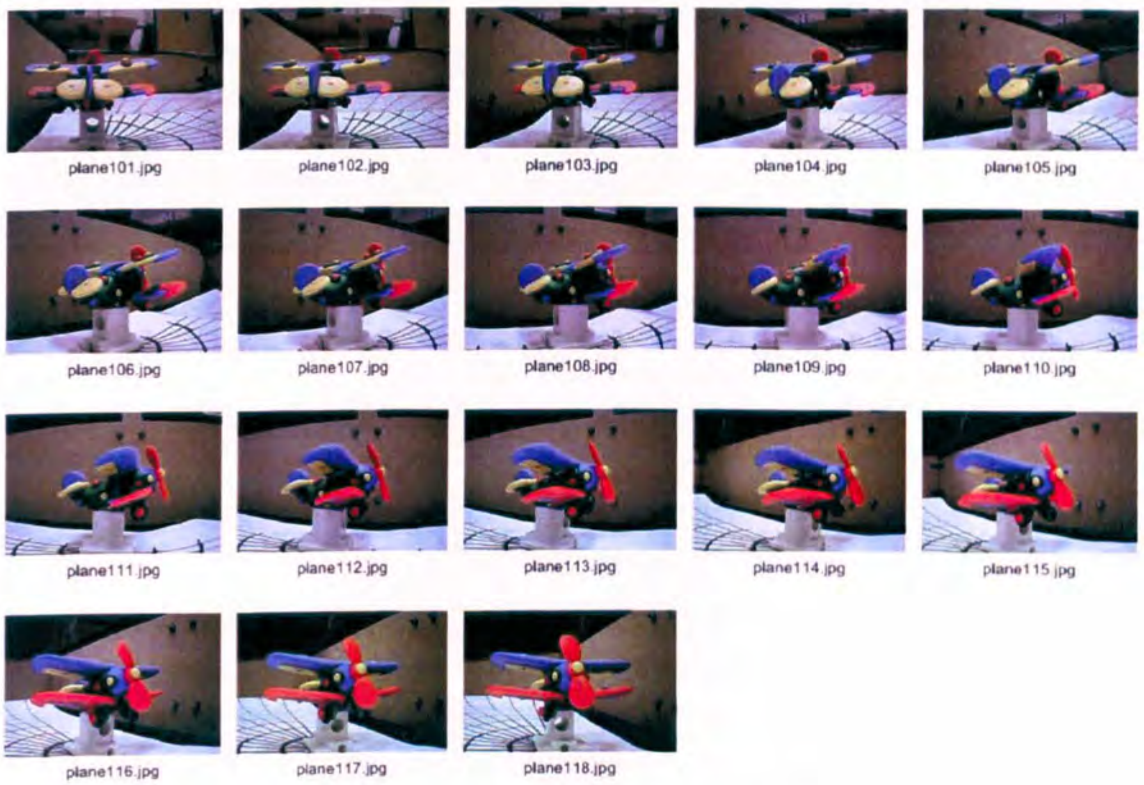


Figure B.3: Test Images Set 1

B.2 Entropy and mutual information

Entropy is a statistical measure of randomness that can be used to characterise the texture of the input image. As can be seen in Figure B.7 biggest changes are observed at edges between objects. This would therefore imply that entropy could also be used as an edge detector. The entropy outcomes shown in Figure B.7 were generated by using a 9×9 sliding window over the figure on the left hand side. For each window, the entropy is then calculated calculated using B.2:

Figure B.4: Test Images Set 2

$$E(x) = -\sum_{i=0}^{N} log_2 \left[\frac{N_i}{N_T} \right] \tag{B.2}$$

where i ranges between the possible greyscale values $[0, N]$. N_i is the number of pixels with values i within a window and N_T is the total number of pixels within that window.

Given two discrete random variables X and Y, the entropy definition can be extended to compute the joint entropy $H(X, Y)$ and the conditional entropy $H(X|Y)$ or $H(Y|X)$ [42]. For instance, the entropy $H(X|Y)$ is the entropy of X if Y is given. The joint entropys chain rule theorem states

$$\begin{aligned} H(X, Y) &= H(X) + H(Y|X) \\ H(X, Y) &= H(Y) + H(X|Y) \\ H(X)H(X|Y) &= H(Y)H(Y|X) \end{aligned}$$

which means that joint entropy is the entropy of one variable plus the conditional entropy of the other. Given that X and Y are statistically independent random varialbes if $p(x, y) = p(x)p(y)$, the following inequalities can be proved:

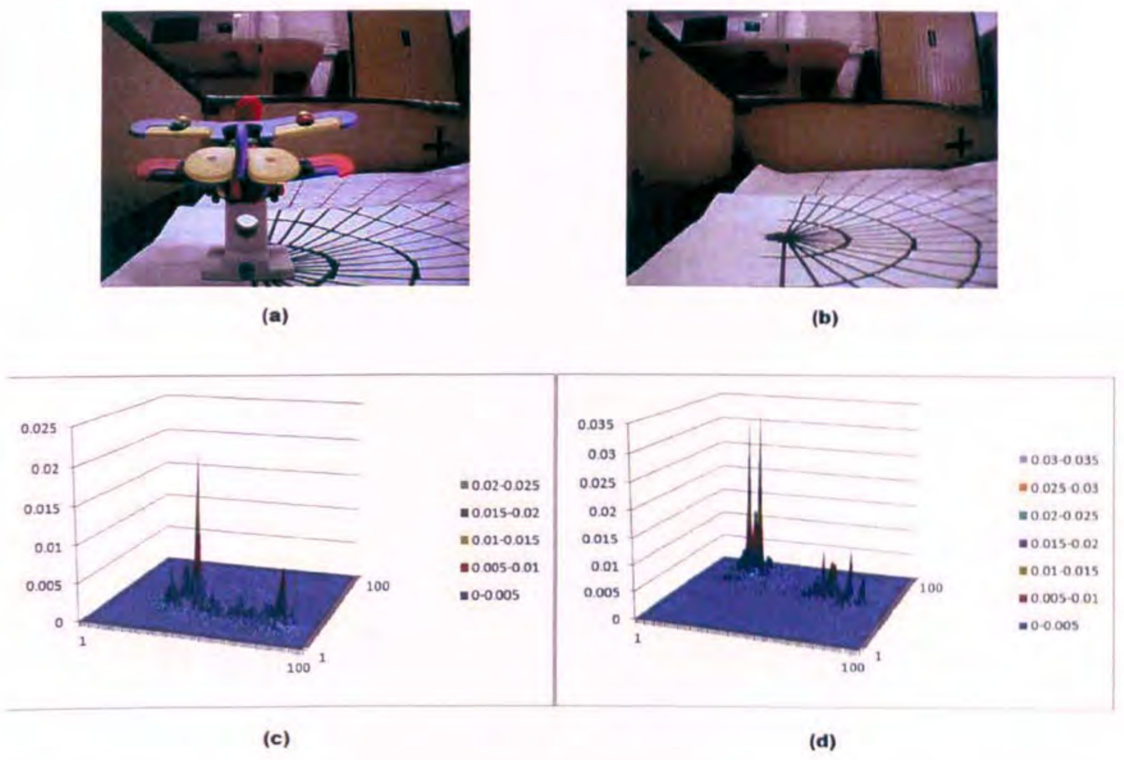


Figure B.5: a) Image 1, b) Image2, c) U vs V PDF of Image 1, d) U vs V PDF of Image 2

$$H(X,Y) \leq H(X) + H(Y)$$
$$H(X|Y) \leq H(X)$$
$$H(Y|X) \leq H(Y)$$

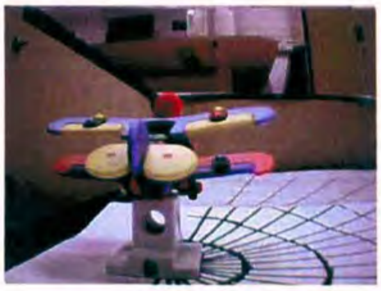
Equalities occur when X and Y are independent random variables.

Mutual information measures the degree to which knowledge of the value of one variable predicts the value of another. If it is zero, then the two variables are independent and otherwise if very large [269]. Put in other words, mutual information is the amount that the uncertainty in Y (or X) is reduced when X (or Y) is known [42] and it can be defined as:

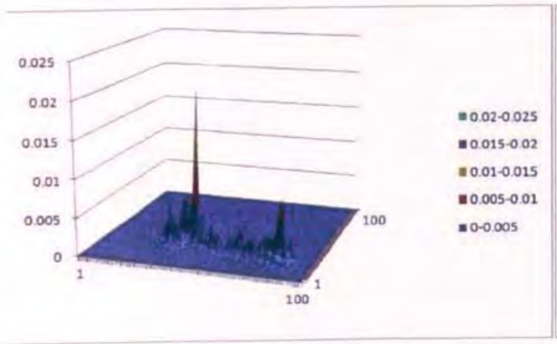
$$I(X;Y) = H(X)H(X|Y) = H(Y)H(Y|X) \quad (B.3)$$
$$I(X;Y) = H(X) + H(Y)H(X,Y). \quad (B.4)$$

Equation B.3 suggests that mutual information may be viewed as a measure of the

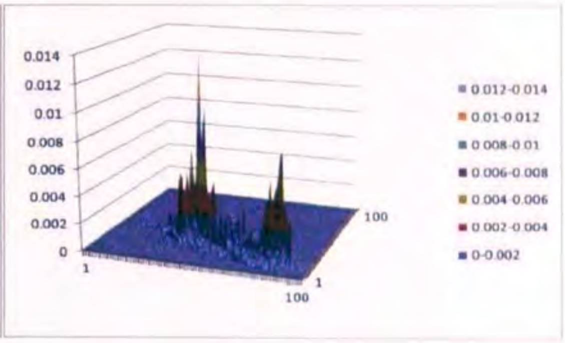
(a)



(b)



(c)



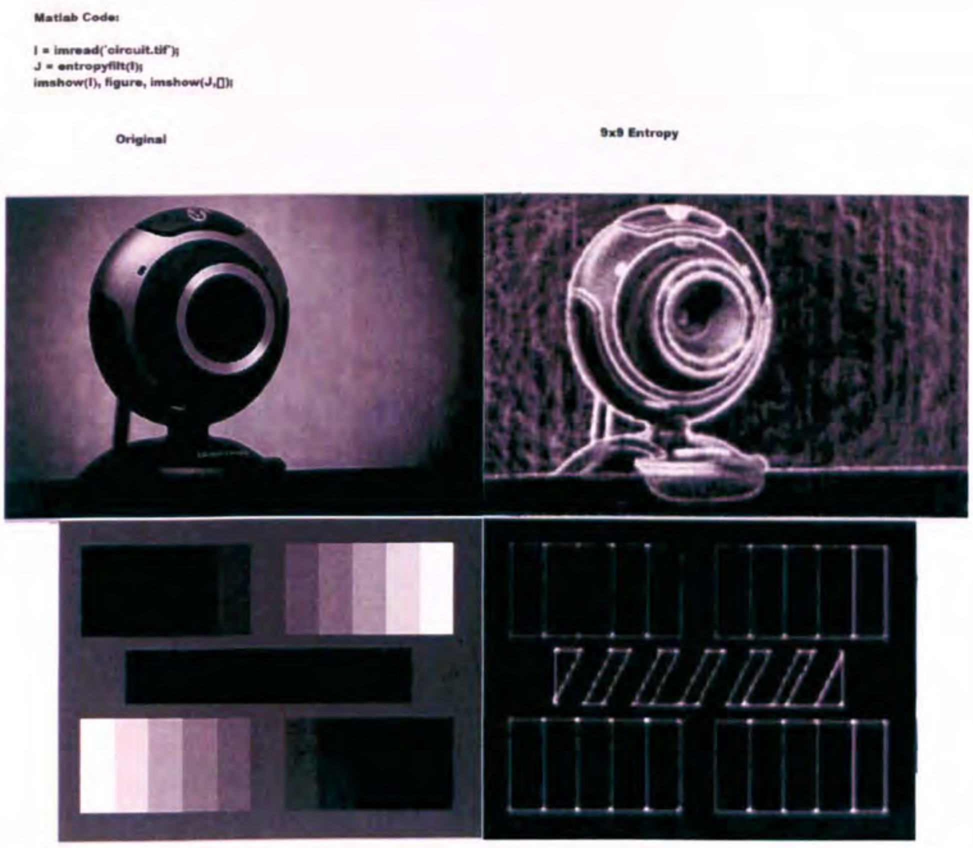
(d)

Figure B.6: a) Image 1, b) Image2, c) U vs V PDF of Image 1, d) U vs V PDF of Image 2

statistical dependence between two random variables. The definitions provided by B.4 states that mutual information is the information of a variable minus its information if the other is given. Note that $I(X;Y) = I(Y;X)$ and $I(X;Y) \geq 0$, where the equality occurs if X and Y are statistically independent random variables. Since $I(X;X) = H(X)H(X|X) = H(X)$, entropy is sometimes referred to as self-information. The conditional mutual information of two random variables X and Y given another random variable Z is defined as $I(X;Y|Z) = H(X|Z)H(X|Y,Z)$ which is a generalization of Equation B.3 to conditional distributions [42].

B.3 Kullback-Leibler divergence

The Kullback-Leibler divergence is a fundamental equation of information theory that quantifies the proximity of two probability distributions, written as shown in Equation (B.5).



Matlab Code:

```
I = imread('circuit.tif');
J = entropyfilt(I);
imshow(I), figure, imshow(J,[]);
```

Figure B.7: Entropy Filter

$$\begin{aligned} D_{KL}(X||Y) &= -\sum_i x(i)\log y(i) + \sum_i x(i)\log x(i) \\ &= H(X,Y) + H(X) \end{aligned} \qquad (B.5)$$

where $H(X,Y)$ is called the cross entropy of X and Y, and $H(X)$ is the entropy of X. The Kullback-Leibler divergence is:

- Always non negative
- Not symmetric
- 0 if distributions match exactly

When working using the Kullback-Leibler divergence particular attention has to be made on PDFs having zero values in any of its bins since this divergence is based on logs and a log of zero would lead to infinity. As indicated in [284] the convention for $0\log(0/q) = 0$, $0\log(0/0) = 0$ and $p\log(p/0) = \infty$. In order to avoid encountering the latter case,for bins with 0 value let value be a very small number such as 1×10^{-7} otherwise this would lead to an inconclusive answer (divide by 0). Although this introduces an error, for practical purposes the results obtained were found to be good enough and sufficient.

Since Kullback-Leibler is not symmetric, ie. $D_{KL}(X||Y) \neq D_{KL}(Y||X)$, both are thus calculated and the output is the average of both. Therefore the output from the matching algorithm would be as indicated in Equation (B.6).

$$\frac{D_{KL}(X||Y) + D_{KL}(Y||X)}{2} \tag{B.6}$$

C
The GNG algorithm

GNG was originally introduced by Fritzke [181], as an unsupervised learning technique where no prior training is needed. The system starts with two linked nodes; new nodes are inserted at every fixed number of input cycles up until the maximum number of allowed nodes is reached. Connections between nodes are also inserted and removed adapting the network topology. Moreover, nodes which are disconnected are removed thus allowing for new nodes to be inserted in a better position within the topological map. This results in a network having a topological structure composed of N nodes in Y clusters connected by edges closely reflecting the topology of the feature distribution.

The GNG network is specified as:

- A set N of nodes (neurons). Each node $k \in N$ has its associated reference vector $\mathbf{w}_k$ belonging to the input space (in this case greyscale images obtained from the video stream).

- A set of edges (connections) between pairs of nodes. These connections are not weighted and its purpose is to define the topological structure. An edge ageing scheme is used to remove connections that are invalid due to the adaptation of the node during the learning process.

The GNG algorithm operates as follows:

(a) Start with two nodes containing random values, connected with a zero age edge and set their errors to zero.

(b) Generate an input vector $\mathbf{x}$.

(c) Find the nearest node s_1 and second nearest node s_2 to $\mathbf{x}$ according to the Euclidean distance, $s_1, s_2 \in N$.

(d) Increase the age of all the edges emanating from s_1.

(e) Update the winner node s_1 error by adding the squared euclidean distance between the input signal and its reference vector.

$$error_{s_1} \leftarrow error_{s_1} + \|\mathbf{w}_{s_1} - \mathbf{x}\|^2$$

(f) Adapt the reference vectors of the winner node s_1 and its topological neighbours towards $\mathbf{x}$ by a learning step ε_{s_1} and ε_n, respectively, of the distance.

$$\begin{aligned} \mathbf{w}_{s_1} &\leftarrow \mathbf{w}_{s_1} + \varepsilon_{s_1}(\mathbf{x} - \mathbf{w}_{s_1}) \\ \mathbf{w}_n &\leftarrow \mathbf{w}_n + \varepsilon_n(\mathbf{x} - \mathbf{w}_n) \end{aligned}$$

$\forall n \in Neighbour(s)$, $\varepsilon_{s_1}, \varepsilon_n \in [0,1]$

(i) If s_1 and s_2 are connected by an edge, then set its age to 0. If they are not connected, then create an edge between them.

(j) If there are any edges with an age larger than a_{max} then remove them. If this results in isolated nodes (without emanating edges) remove them as well.

(k) Every λ iterations and if the total size of the network has not been reached then insert a new node as follows:

- Find the node u with largest error.
- Among neighbours of u, find the node v with largest error
- Insert a new node r between u and v using

$$\mathbf{w}_r \leftarrow \frac{\mathbf{w}_u + \mathbf{w}_v}{2}$$

- Create edges between u and r, and v and r. Remove edge between u and v.
- Decrease the error variables of u and v, by α_u and α_v which are negative and set the error of node r.

$$\begin{aligned} error_u &\leftarrow error_u + \alpha_u \\ error_v &\leftarrow error_v + \alpha_v \\ error_r &\leftarrow -(\alpha_u + \alpha_v) \end{aligned}$$

where

$$\alpha_u = -0.5 \left(\frac{error_u}{error_u + error_v} \right) error_u$$

$$\alpha_v = -0.5 \left(1 - \frac{error_u}{error_u + error_v} \right) error_v$$

(l) Decrease all error variables of all nodes $j \in N$ by a factor β.

$$error_j \leftarrow error_j - \beta \times error_j$$

(m) If the stopping criterion is not met then repeat from step (b).

D
Robot Configuration

D.1 Hardware

Several mobile platforms were used throughout the course of this study. The robot vision team (RoViT), part of the Digital Imaging Research Centre (DIRC) in the Faculty of Science, Engineering and Technology (SEC) owns several robotic platforms. Figure D.1 shows some of them, namely the Pioneer[1] (back left), Videre Erratic[2] (back right), Rovio[3] (middle) and Surveyor[4] (front). Amongst these a humanoid robot is also available but which was not used for the course of this study. The Pioneer robot platform is the most rugged professional robot available within our fleet, capable of hosting multiple sensors and onboard computing. The Erratic robots are a cheaper version of the Pioneer designed by Videre but which are still good enough for testing / research purposes. The other small robots are very basic platforms, with very limited capabilities, for example, no effective sensing, single pinhole cameras and basic mobile functionality. During hardware selection it was concluded that the limited capabilities of such robots, prohibit experimentation with complex vision and machine learning algorithms. The inferior onboard camera also makes image processing difficult and error prone. As a result they are unsuitable for proper experimentation and they cannot be used to take part in public demonstration or competitions.

[1]http://www.mobilerobots.com/researchrobots/pioneerp3dx.aspx
[2]http://www.videre.com (possibly discontinued)
[3]http://www.wowwee.com/en/support/rovio/
[4]http://www.surveyor.com/SRV info.html

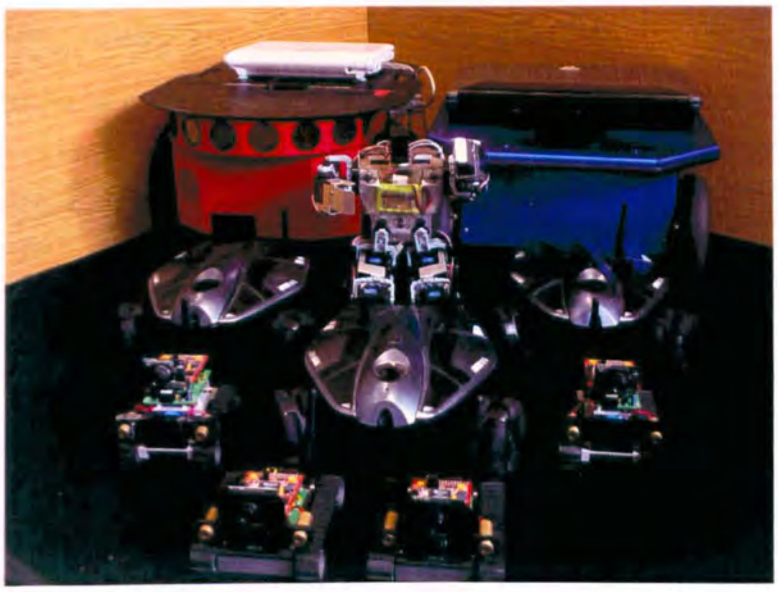


Figure D.1: The fleet

D.1.1 Robot base

Most of our experimentation was carried out using the the Erratic (ERA) mobile robot. This platform is a full-featured, industrial mobile robot base. The name comes from the Latin errare: to wander. The ERA is compact and powerful platform, and capable of carrying a full load of robotics equipment, including an integrated PC, laser rangefinder, and stereo cameras [285]. Its specifications are shown in Table D.1.

Base platform size	40 cm (L) x 37 cm (W) x 18 cm (H)
Wheels	15 cm diameter (driven) 6.25 cm diameter (caster) Polymer core, soft non-marking rubber tread
Wheelbase	33 cm
Drive type	Differential, single rear caster
Maximum speed	2.0 m/sec, 720 deg/sec
Motors	DC reversible with gearhead 72 W continuous power
Encoder resolution	500 cycles per motor revolution
Controller	16 bit microcontroller Integrated controller / motor driver Analog, digital, and servo interfaces
Power	12V, 7AH lead-acid batteries (x3) 5A charger
Weight	4.5 Kg (base) 12 Kg (base + 3 batteries)
Payload	20 Kg

Table D.1: Erratic Robot Base Specifications

D.1.2 Laser scanner

A laser range finder is a device which uses a laser beam to determine the distance to an object. The most common form of laser range finder operates on the time of flight principle by sending a laser pulse in a narrow beam towards the object and measuring the time taken by the pulse to be reflected off the target and returned to the sender. Figures D.2 and D.3 show an image of the actual laser rangefinders used with their respective specifications shown in Tables D.2 and D.3.

Figure D.2: Hokuyo Laser 1

Specifications	
Voltage	5.0 V ± 5 %
Current	0.5 A (Rush current 0.8 A)
Detection Range	0.02 m to approximately 4 m
Laser wavelength	785 nm, Class 1
Scan angle	240 °
Scan time	100 ms/scan (10.0 Hz)
Resolution	1 mm
Accuracy	Distance 20 ~ 1000 mm: ±10 mm
	Distance 1000 ~ 4000 mm: ±1 % of measurement
Angular Resolution	0.36 °
Interface	USB 2.0, RS232
Weight	141 gm (5.0 oz)

Table D.2: Hokuyo Laser 1 Specs

Figure D.3: Hokuyo Laser 2

Specifications	
Voltage	5.0 V ± 5 %
Current	0.5 A nominal (rush current 0.8 A)
Detection Range	20 mm (0.79 in) to ~ 5.6 m (18.37 ft); < 4 m (13 ft) guaranteed
Laser wavelength	785 nm, Class 1
Scan angle	240 °
Scan time	100 msec/scan (10.0 Hz)
Resolution	1 mm
Accuracy	Distance 20 mm ~ 1000 mm : ±30 mm
	Distance 20 mm ~ 4000 mm : ±3 % of measurement
Angular resolution	0.36 °
Interface	USB 2.0
Weight	5.64 oz (160 g)

Table D.3: Hokuyo Laser 2 Specs

D.1.3 Kinect sensor

Kinect, shown in Figure D.4 is a device initially developed for the Microsoft Xbox 360 by PrimeSense and released in November 2010. It is composed of an RGB camera, 3D depth sensors, a multi-array microphone and a motorized tilt.

The main characteristics of the Kinect include:

- An RGB sensor. This is a regular camera that streams video with 8 bits for every color channel, giving a 24-bit color depth. Its color filter array is a Bayer filter mosaic and the color resolution is 640 × 480 pixels with a maximal frame rate of 30 Hz.

[5] http://blog.robotiq.com/bid/40428/Using-The-Kinect-For-Robotic-Manipulation

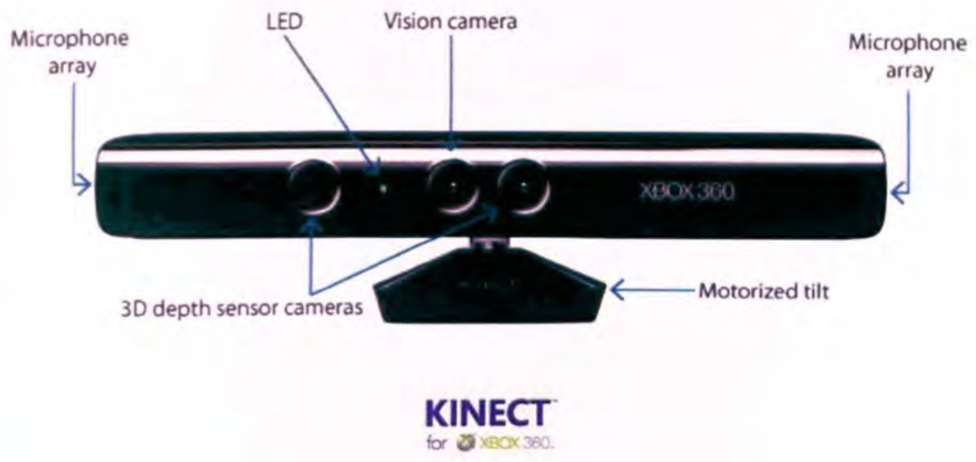


Figure D.4: Kinect Sensor[5]

- A depth sensing system composed of an IR emitter projecting structured light, which is captured by the CMOS image sensor, and decoded to produce the depth image of the scene. Its range is specified to be between 0.7 and 6 meters, although the best results are obtained from 1.2 to 3.5 meters. Its data output has 12-bit depth. The depth sensor resolution is 320 × 240 pixels with a rate of 30 Hz.
- A field of viev of 57^o horizontal, 43^o vertical, with a tilt range of $\pm 27^o$.

D.2 Software - ROS robot setup

In this section, the robot setup within ROS is presented.

D.2.1 Transform tree

The transform tree (tf) package within ROS is a package which lets the user keep track of multiple coordinate frames over time. This package maintains the relationship between coordinate frames in a tree structure buffered in time, and lets the user transform points, vectors, etc. between any two coordinate frames at any desired point in time. Figure D.5 shows the transform tree for our robot configuration, linking the erratic base platform, odometry encoders, a Kinect and a laser range finder.

D.2.2 ROS launch files

Launch files are in one or more XML configuration files (with the .launch extension) that specify the parameters to set and nodes to launch, as well as the machines that they should be run on. This section contains the launch files required to operate our robots.

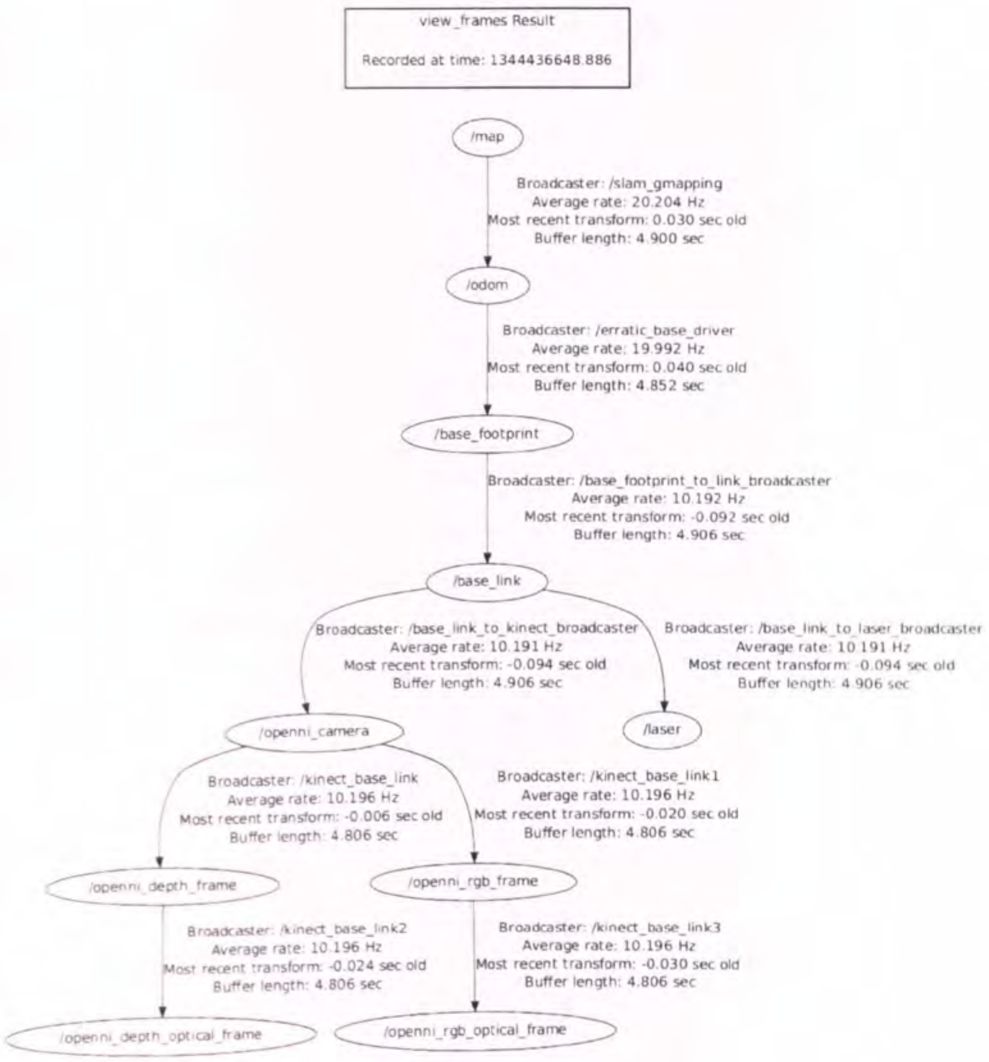

view_frames Result
Recorded at time: 1344436648.886
/map
Broadcaster: /slam_gmapping
Average rate: 20.204 Hz
Most recent transform: 0.030 sec old
Buffer length: 4.900 sec
/odom
Broadcaster: /erratic_base_driver
Average rate: 19.992 Hz
Most recent transform: 0.040 sec old
Buffer length: 4.852 sec
/base_footprint
Broadcaster: /base_footprint_to_link_broadcaster
Average rate: 10.192 Hz
Most recent transform: -0.092 sec old
Buffer length: 4.906 sec
/base_link
Broadcaster: /base_link_to_kinect_broadcaster
Average rate: 10.191 Hz
Most recent transform: -0.094 sec old
Buffer length: 4.906 sec
Broadcaster: /base_link_to_laser_broadcaster
Average rate: 10.191 Hz
Most recent transform: -0.094 sec old
Buffer length: 4.906 sec
/openni_camera
/laser
Broadcaster: /kinect_base_link
Average rate: 10.196 Hz
Most recent transform: -0.006 sec old
Buffer length: 4.806 sec
Broadcaster: /kinect_base_link1
Average rate: 10.196 Hz
Most recent transform: -0.020 sec old
Buffer length: 4.806 sec
/openni_depth_frame
/openni_rgb_frame
Broadcaster: /kinect_base_link2
Average rate: 10.196 Hz
Most recent transform: -0.024 sec old
Buffer length: 4.806 sec
Broadcaster: /kinect_base_link3
Average rate: 10.196 Hz
Most recent transform: -0.030 sec old
Buffer length: 4.806 sec
/openni_depth_optical_frame
/openni_rgb_optical_frame

Figure D.5: Robot Transform Tree

dhs_laser_navigation.launch

```
<launch>
    <include file="$(find openni_launch)/launch/openni.launch"/>
    <include file="$(find DHS_DEMO)/launch/
                    dhs_robot_laser_configuration.launch"/>
    <include file="$(find DHS_DEMO)/launch/dhs_move_base.launch"/>

    <node pkg="TCPInterface" type="tcpinterface"
    name="TCPInterface" output="screen">
        <param name="TCP_server_port" type="int" value="9000" />
        <param name="UDP_server_port" type="int" value="0" />
</node>
```

```
    <node pkg="DHS_DEMO" type="TCPActionServer.py"
    name="TCPActionServer" output="screen" />
</launch>
```

dhs_robot_laser_configuration.launch

```
<launch>
    <node pkg="tf" type="static_transform_publisher"
    name="base_link_to_kinect_broadcaster"
    args="-0.115 0 0.226 0 0 0 base_link openni_camera 100" />

    <node pkg="tf" type="static_transform_publisher"
    name="base_footprint_to_link_broadcaster"
    args="0 0 0 0 0 0 base_footprint base_link 100" />

    <node pkg="tf" type="static_transform_publisher"
    name="base_link_to_laser_broadcaster"
    args="0.050 0 0.300 0 0 0 base_link laser 100" />

    <node name="hokuyo" pkg="hokuyo_node" type="hokuyo_node"
    respawn="false" output="screen">

        <!-- Starts up faster, but timestamps will be inaccurate. -->
        <param name="calibrate_time" type="bool" value="false"/>

        <!-- Set the port to connect to here -->
        <param name="port" type="string" value="/dev/ttyACM0"/>

        <param name="intensity" type="bool" value="false"/>
    </node>

    <node name="erratic_base_driver" pkg="erratic_player"
    type="erratic_player" output="screen" >
        <remap from="odom" to="odom"/>
        <remap from="battery_state" to="battery_state"/>
        <param name="port_name" type="str" value="/dev/ttyUSB0"/>
        <param name="enable_ir" type="bool" value="False"/>
        <param name="odometry_frame_id" type="str" value="odom"/>
    </node>
</launch>
```

dhs_move_base.launch

```
<launch>
    <!-- Run the map server -->
    <node name="map_server" pkg="map_server"
    type="map_server" args="$(find DHS_DEMO)/maps/
                                            KUHill/KUHill-2.yaml"/>

    <!--- Run AMCL -->
    <include file="$(find DHS_DEMO)/launch/amcl_diff.launch" />

    <node pkg="move_base" type="move_base"
    respawn="false" name="move_base" output="screen">
        <rosparam file="$(find DHS_DEMO)/info/costmap_common_params.yaml"
        command="load" ns="global_costmap" />
        <rosparam file="$(find DHS_DEMO)/info/costmap_common_params.yaml"
        command="load" ns="local_costmap" />
        <rosparam file="$(find DHS_DEMO)/info/local_costmap_params.yaml"
        command="load" />
        <rosparam file="$(find DHS_DEMO)/info/global_costmap_params.yaml"
        command="load" />
        <rosparam file="$(find DHS_DEMO)/info/base_local_planner_params.yaml'
        command="load" />
    </node>
</launch>
```

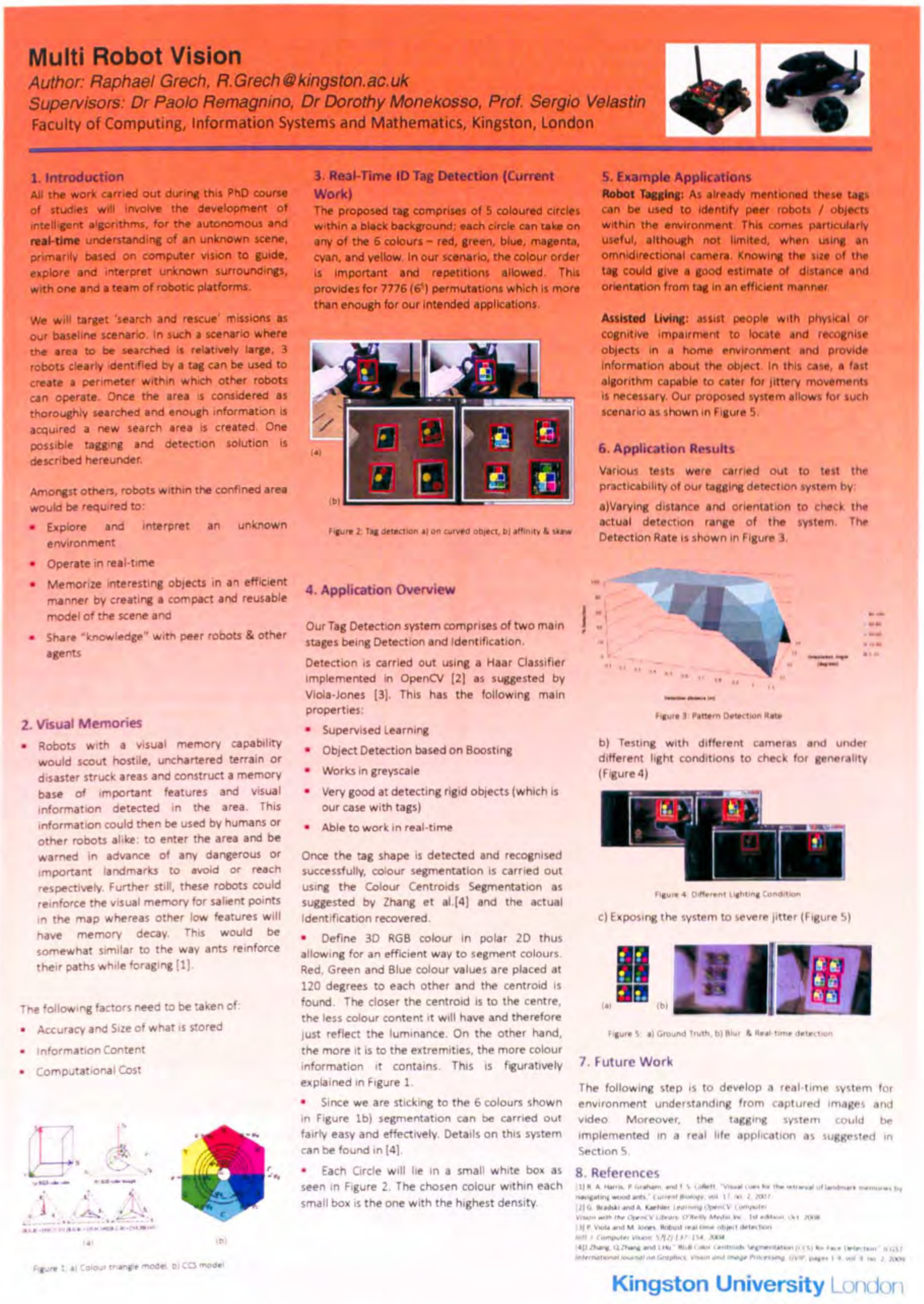
amcl_diff.launch

```
<launch>
    <node pkg="amcl" type="amcl" name="amcl" output="screen">
        <!-- Publish scans from best pose at a max of 10 Hz -->
        <param name="odom_model_type" value="diff"/>
        <param name="odom_alpha5" value="0.1"/>
        <param name="transform_tolerance" value="0.2" />
        <param name="gui_publish_rate" value="10.0"/>
        <param name="laser_max_beams" value="60"/>
        <param name="min_particles" value="1000"/>
        <param name="max_particles" value="10000"/>
        <param name="kld_err" value="0.05"/>
        <param name="kld_z" value="0.99"/>
        <param name="odom_alpha1" value="0.2"/>
        <param name="odom_alpha2" value="0.2"/>
        <!-- translation std dev, m -->
        <param name="odom_alpha3" value="0.8"/>
        <param name="odom_alpha4" value="0.2"/>
        <param name="laser_z_hit" value="0.5"/>
        <param name="laser_z_short" value="0.05"/>
        <param name="laser_z_max" value="0.05"/>
        <param name="laser_z_rand" value="0.5"/>
        <param name="laser_sigma_hit" value="0.2"/>
        <param name="laser_lambda_short" value="0.1"/>
        <param name="laser_lambda_short" value="0.1"/>
        <param name="laser_model_type" value="likelihood_field"/>
        <!-- <param name="laser_model_type" value="beam"/> -->
        <param name="laser_likelihood_max_dist" value="2.0"/>
        <param name="update_min_d" value="0.2"/>
        <param name="update_min_a" value="0.5"/>
        <param name="odom_frame_id" value="odom"/>
        <param name="resample_interval" value="1"/>
        <param name="transform_tolerance" value="0.1"/>
        <param name="recovery_alpha_slow" value="0.0"/>
        <param name="recovery_alpha_fast" value="0.0"/>
    </node>
</launch>
```

E
Posters

This appendix contains the various posters set up during the PhD research period.

- E.1 - tag detection and identification
- E.2 - robot setup and system overview
- E.3 - Saliency and visual memory
- E.4 - Information fusion



Multi Robot Vision

Author: Raphael Grech, R.Grech@kingston.ac.uk
Supervisors: Dr Paolo Remagnino, Dr Dorothy Monekosso, Prof. Sergio Velastin
Faculty of Computing, Information Systems and Mathematics, Kingston, London

1. Introduction

All the work carried out during this PhD course of studies will involve the development of intelligent algorithms, for the autonomous and **real-time** understanding of an unknown scene, primarily based on computer vision to guide, explore and interpret unknown surroundings, with one and a team of robotic platforms.

We will target 'search and rescue' missions as our baseline scenario. In such a scenario where the area to be searched is relatively large, 3 robots clearly identified by a tag can be used to create a perimeter within which other robots can operate. Once the area is considered as thoroughly searched and enough information is acquired a new search area is created. One possible tagging and detection solution is described hereunder.

Amongst others, robots within the confined area would be required to:

- Explore and interpret an unknown environment
- Operate in real-time
- Memorize interesting objects in an efficient manner by creating a compact and reusable model of the scene and
- Share "knowledge" with peer robots & other agents

2. Visual Memories

- Robots with a visual memory capability would scout hostile, unchartered terrain or disaster struck areas and construct a memory base of important features and visual information detected in the area. This information could then be used by humans or other robots alike: to enter the area and be warned in advance of any dangerous or important landmarks to avoid or reach respectively. Further still, these robots could reinforce the visual memory for salient points in the map whereas other low features will have memory decay. This would be somewhat similar to the way ants reinforce their paths while foraging [1].

The following factors need to be taken of:

- Accuracy and Size of what is stored
- Information Content
- Computational Cost

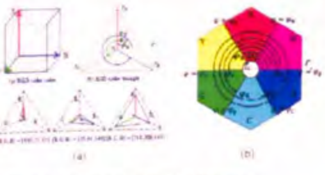


Figure 1: a) Colour triangle model. b) CCS model

3. Real-Time ID Tag Detection (Current Work)

The proposed tag comprises of 5 coloured circles within a black background; each circle can take on any of the 6 colours – red, green, blue, magenta, cyan, and yellow. In our scenario, the colour order is important and repetitions allowed. This provides for 7776 (6^5) permutations which is more than enough for our intended applications.

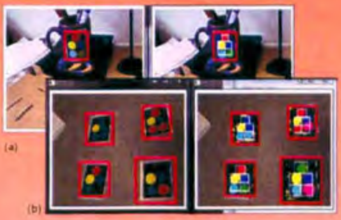


Figure 2: Tag detection a) on curved object, b) affinity & skew

4. Application Overview

Our Tag Detection system comprises of two main stages being Detection and Identification.

Detection is carried out using a Haar Classifier implemented in OpenCV [2] as suggested by Viola-Jones [3]. This has the following main properties:

- Supervised Learning
- Object Detection based on Boosting
- Works in greyscale
- Very good at detecting rigid objects (which is our case with tags)
- Able to work in real-time

Once the tag shape is detected and recognised successfully, colour segmentation is carried out using the Colour Centroids Segmentation as suggested by Zhang et al.[4] and the actual identification recovered.

- Define 3D RGB colour in polar 2D thus allowing for an efficient way to segment colours. Red, Green and Blue colour values are placed at 120 degrees to each other and the centroid is found. The closer the centroid is to the centre, the less colour content it will have and therefore just reflect the luminance. On the other hand, the more it is to the extremities, the more colour information it contains. This is figuratively explained in Figure 1.
- Since we are sticking to the 6 colours shown in Figure 1b) segmentation can be carried out fairly easy and effectively. Details on this system can be found in [4].
- Each Circle will lie in a small white box as seen in Figure 2. The chosen colour within each small box is the one with the highest density.

5. Example Applications

Robot Tagging: As already mentioned these tags can be used to identify peer robots / objects within the environment. This comes particularly useful, although not limited, when using an omnidirectional camera. Knowing the size of the tag could give a good estimate of distance and orientation from tag in an efficient manner.

Assisted Living: assist people with physical or cognitive impairment to locate and recognise objects in a home environment and provide information about the object. In this case, a fast algorithm capable to cater for jittery movements is necessary. Our proposed system allows for such scenario as shown in Figure 5.

6. Application Results

Various tests were carried out to test the practicability of our tagging detection system by:

a)Varying distance and orientation to check the actual detection range of the system. The Detection Rate is shown in Figure 3.

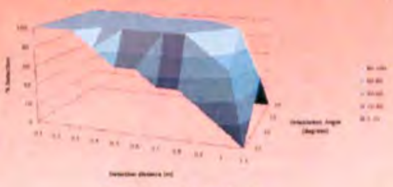


Figure 3: Pattern Detection Rate

b) Testing with different cameras and under different light conditions to check for generality (Figure 4)

Figure 4: Different Lighting Condition

c) Exposing the system to severe jitter (Figure 5)

Figure 5: a) Ground Truth, b) Blur & Real-time detection

7. Future Work

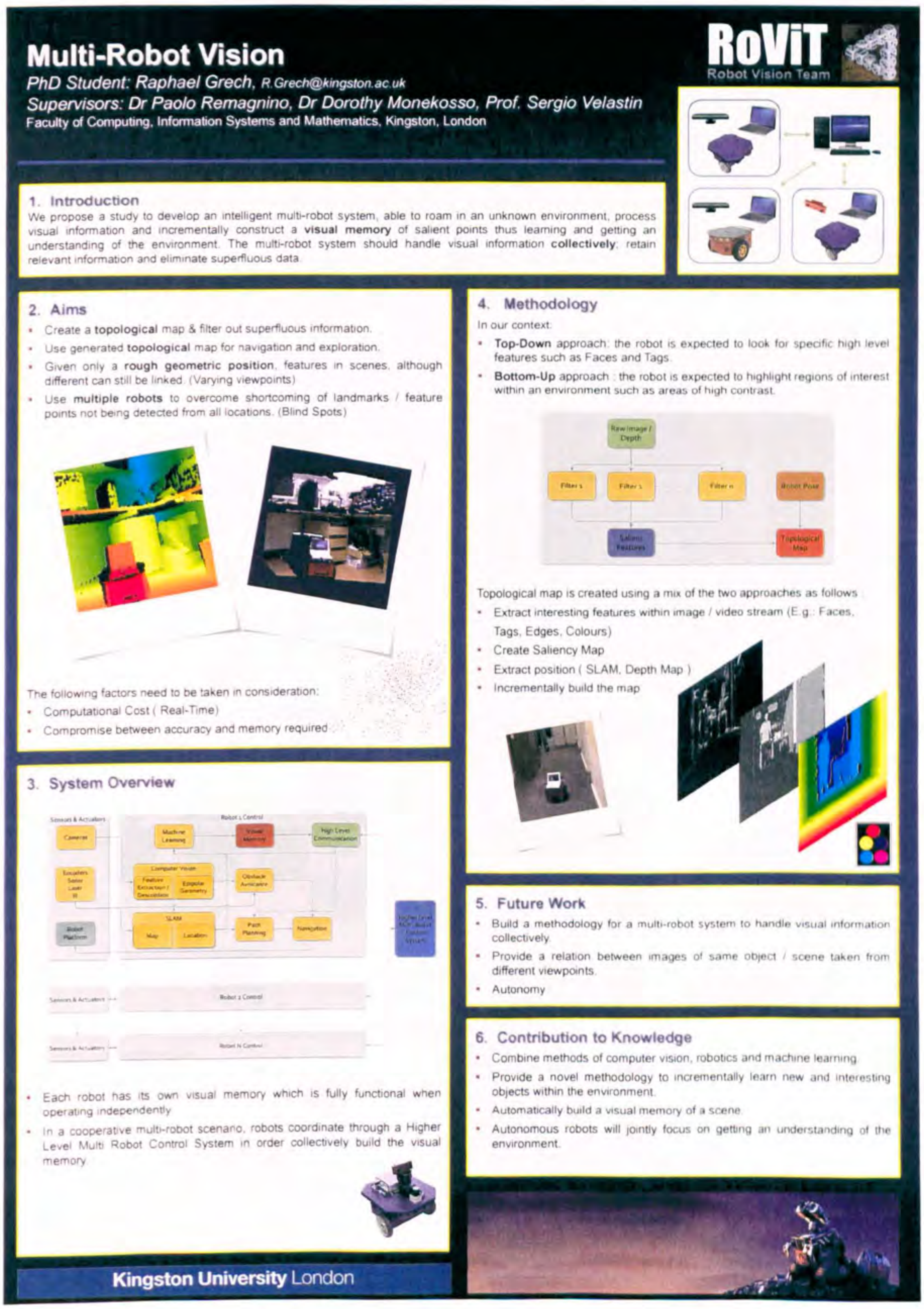
The following step is to develop a real-time system for environment understanding from captured images and video. Moreover, the tagging system could be implemented in a real life application as suggested in Section 5.

8. References

[1] R. A. Harris, P. Graham, and T. S. Collett. "Visual cues for the retrieval of landmark memories by navigating wood ants," *Current Biology*, vol. 17, no. 2, 2007.
[2] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library.* O'Reilly Media, Inc., 1st edition, Oct. 2008
[3] P. Viola and M. Jones. Robust real-time object detection. *Intl. J. Computer Vision*, 57(2):137–154, 2004
[4] D. Zhang, Q. Zhang and J. Hu " RGB Color Centroids Segmentation (CCS) for Face Detection" ICGST *International Journal on Graphics, Vision and Image Processing*, GVIP, pages 1-9, vol. 9, no. 2, 2009

Kingston University London

Figure E.1: BMVA Summer School 2010



Multi-Robot Vision

PhD Student: Raphael Grech, R.Grech@kingston.ac.uk
Supervisors: Dr Paolo Remagnino, Dr Dorothy Monekosso, Prof. Sergio Velastin
Faculty of Computing, Information Systems and Mathematics, Kingston, London

RoViT
Robot Vision Team

1. Introduction

We propose a study to develop an intelligent multi-robot system, able to roam in an unknown environment, process visual information and incrementally construct a **visual memory** of salient points thus learning and getting an understanding of the environment. The multi-robot system should handle visual information **collectively**, retain relevant information and eliminate superfluous data.

2. Aims

- Create a **topological** map & filter out superfluous information.
- Use generated **topological** map for navigation and exploration.
- Given only a **rough geometric position**, features in scenes, although different can still be linked. (Varying viewpoints)
- Use **multiple robots** to overcome shortcoming of landmarks / feature points not being detected from all locations. (Blind Spots)

The following factors need to be taken in consideration:

- Computational Cost (Real-Time)
- Compromise between accuracy and memory required

3. System Overview

- Each robot has its own visual memory which is fully functional when operating independently
- In a cooperative multi-robot scenario, robots coordinate through a Higher Level Multi Robot Control System in order collectively build the visual memory

4. Methodology

In our context:

- **Top-Down** approach: the robot is expected to look for specific high level features such as Faces and Tags.
- **Bottom-Up** approach : the robot is expected to highlight regions of interest within an environment such as areas of high contrast.

Topological map is created using a mix of the two approaches as follows :

- Extract interesting features within image / video stream (E.g.: Faces, Tags, Edges, Colours)
- Create Saliency Map
- Extract position (SLAM, Depth Map)
- Incrementally build the map

5. Future Work

- Build a methodology for a multi-robot system to handle visual information collectively.
- Provide a relation between images of same object / scene taken from different viewpoints.
- Autonomy

6. Contribution to Knowledge

- Combine methods of computer vision, robotics and machine learning.
- Provide a novel methodology to incrementally learn new and interesting objects within the environment.
- Automatically build a visual memory of a scene.
- Autonomous robots will jointly focus on getting an understanding of the environment.

Kingston University London

Figure E.2: Poster Event 2011

Robot Visual Memory Maps : compact encoding of Saliency

PhD Student: Raphael Grech, R.Grech@kingston.ac.uk
Supervisors: Dr Paolo Remagnino, Dr Dorothy Monekosso
Faculty of Computing, Information Systems and Mathematics, Kingston, London

RoViT
Robot Vision Team

1. Introduction

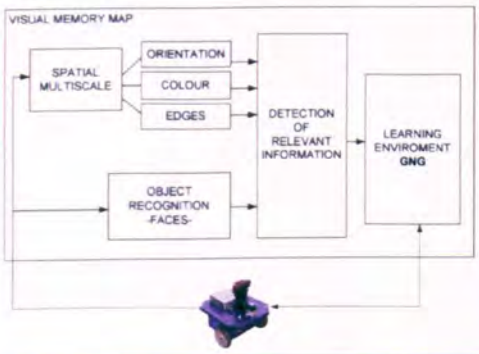
We propose a method to incrementally build a visual memory for a mobile robotic platform. Images captured during robot exploration of an unchartered environment are converted into saliency maps and in turn compactly stored in a neural network. Saliency is a combination of task-oriented search (top-down) and bottom-up visual information based on colour, gradient and orientation of the captured visual field.

2. Visual Memory Model

- We try to mimic human visual attention.
- Saliency Images are extracted from environment images are fed into a Growing Neural Gas (**GNG**), evolving into separate clusters.

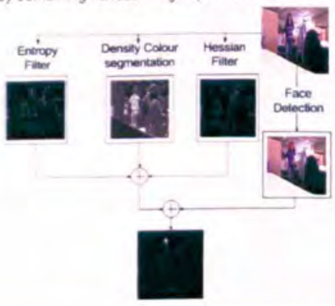
In our context:

- **Top-Down** approach: the robot is expected to look for specific high level features such as Faces.
- **Bottom-Up** approach: the robot is expected to highlight regions of interest within an environment such as areas of high colour contrast or sharp edges.



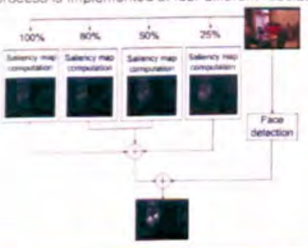
3. Detection of Saliency Map

- Obtained by combining various image operators



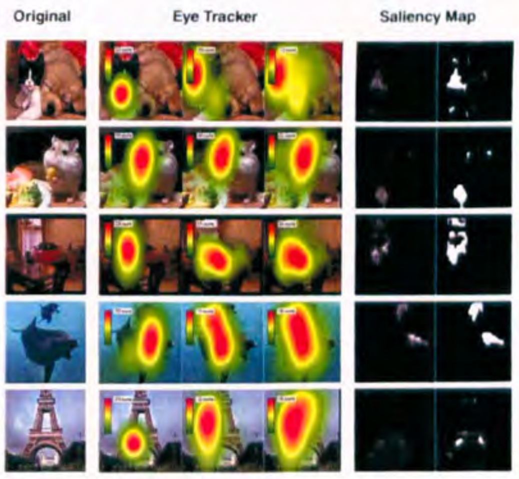
4. Multi-scale Process

- The above process is implemented at four different resolutions



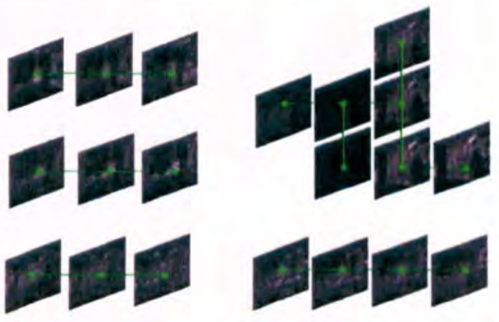
5. Results

- Saliency maps generated by our algorithm are compared with those provided by an eye tracker (3 subjects).



6. Generated Visual Memory Topology

- Saliency maps fed into learning network & nodes grouped in clusters



7. Conclusions

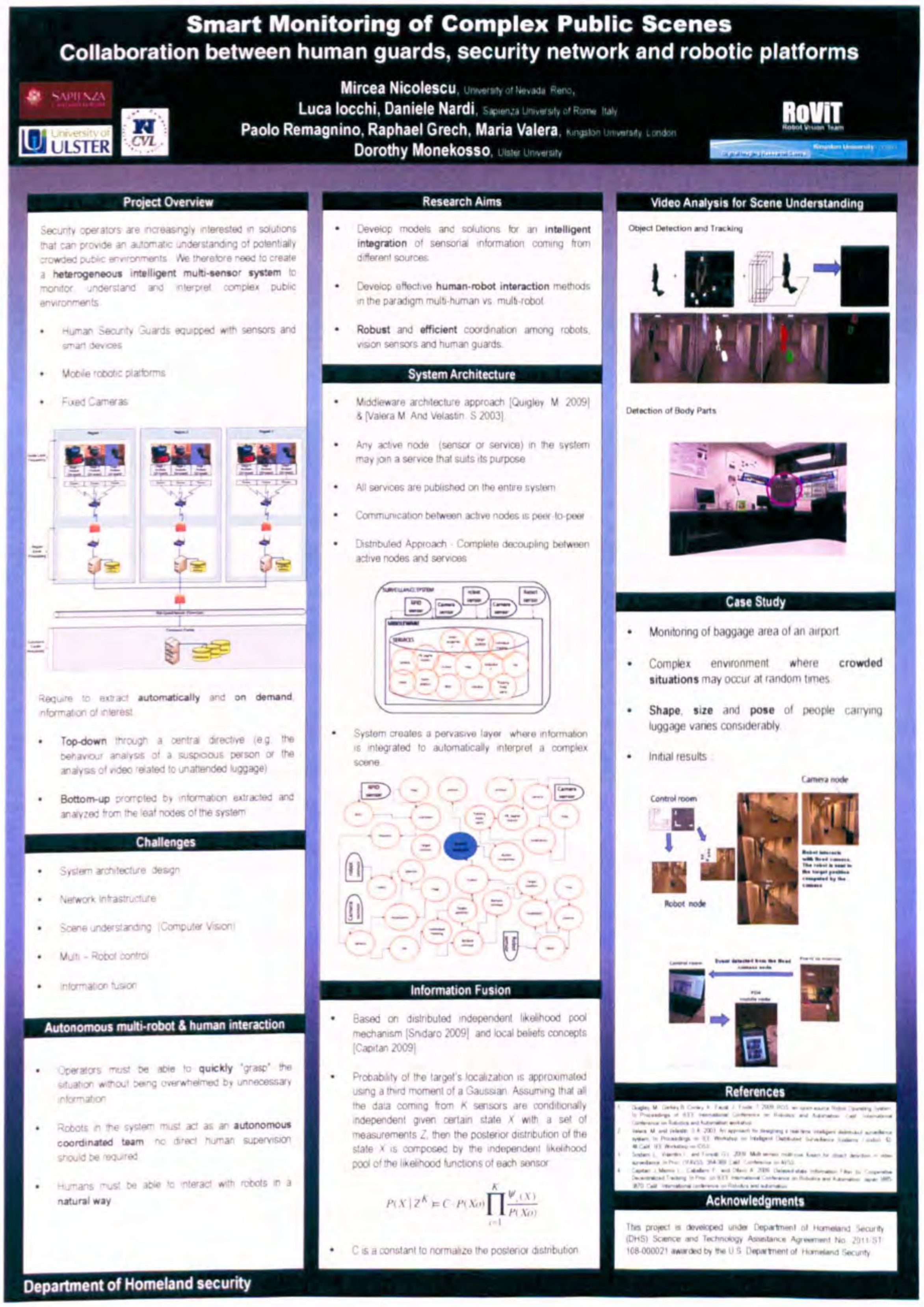
- Our visual attention algorithm results closely resembles those of humans.
- Our system successfully creates and clusters a dynamic memory database

8. Future Work

- Linking of semantic map with geometrical map obtained from the robot and
- Use saliency points to guide the robot to explore that region of interest

Kingston University London

Figure E.3: Video SurveillanceEvent 2011



Smart Monitoring of Complex Public Scenes
Collaboration between human guards, security network and robotic platforms

Mircea Nicolescu, University of Nevada Reno,
Luca Iocchi, Daniele Nardi, Sapienza University of Rome Italy
Paolo Remagnino, Raphael Grech, Maria Valera, Kingston University London
Dorothy Monekosso, Ulster University

Project Overview

Security operators are increasingly interested in solutions that can provide an automatic understanding of potentially crowded public environments. We therefore need to create a **heterogeneous intelligent multi-sensor system** to monitor, understand and interpret complex public environments.

- Human Security Guards equipped with sensors and smart devices
- Mobile robotic platforms
- Fixed Cameras

Require to extract **automatically** and **on demand**, information of interest

- **Top-down** through a central directive (e.g. the behaviour analysis of a suspicious person or the analysis of video related to unattended luggage)
- **Bottom-up** prompted by information extracted and analyzed from the leaf nodes of the system

Challenges

- System architecture design
- Network Infrastructure
- Scene understanding (Computer Vision)
- Multi – Robot control
- Information fusion

Autonomous multi-robot & human interaction

- Operators must be able to **quickly** "grasp" the situation without being overwhelmed by unnecessary information
- Robots in the system must act as an **autonomous coordinated team**: no direct human supervision should be required
- Humans must be able to interact with robots in a **natural way**

Research Aims

- Develop models and solutions for an **intelligent integration** of sensorial information coming from different sources
- Develop effective **human-robot interaction** methods in the paradigm multi-human vs. multi-robot
- **Robust** and **efficient** coordination among robots, vision sensors and human guards.

System Architecture

- Middleware architecture approach [Quigley M 2009] & [Valera M. And Velastin. S 2003]
- Any active node (sensor or service) in the system may join a service that suits its purpose
- All services are published on the entire system
- Communication between active nodes is peer-to-peer
- Distributed Approach - Complete decoupling between active nodes and services
- System creates a pervasive layer where information is integrated to automatically interpret a complex scene

Information Fusion

- Based on distributed independent likelihood pool mechanism [Snidaro 2009] and local beliefs concepts [Capitan 2009]
- Probability of the target's localization is approximated using a third moment of a Gaussian. Assuming that all the data coming from K sensors are conditionally independent given certain state X with a set of measurements Z, then the posterior distribution of the state X is composed by the independent likelihood pool of the likelihood functions of each sensor

$$P(X \mid Z^K) = C \cdot P(Xo) \prod_{i=1}^{K} \frac{\Psi_i(X)}{P(Xo)}$$

- C is a constant to normalize the posterior distribution

Video Analysis for Scene Understanding

Object Detection and Tracking

Detection of Body Parts

Case Study

- Monitoring of baggage area of an airport.
- Complex environment where **crowded situations** may occur at random times.
- **Shape**, **size** and **pose** of people carrying luggage varies considerably.
- Initial results :

References

1. Quigley M. Gerkey B. Conley K. Faust J. Foote T. 2009. ROS: an open-source Robot Operating System. In Proceedings of IEEE International Conference on Robotics and Automation. Call: International Conference on Robotics and Automation workshop.
2. Valera M. and Velastin S.A. 2003. An approach for designing a real-time intelligent distributed surveillance system. In Proceedings on IEE Workshop on Intelligent Distributed Surveillance Systems. London. 42-48. Call: IEE Workshop on IDSS.
3. Snidaro L. Visentini I. and Foresti G.L. 2009. Multi-sensor multi-cue fusion for object detection in video surveillance. In Proc. of AVSS. 364-369. Call: Conference on AVSS.
4. Capitan J. Merino L. Caballero F. and Ollero A. 2009. Delayed-state Information Filter for Cooperative Decentralized Tracking. In Proc. on IEEE International Conference on Robotics and Automation. Japan 3865-3870. Call: International conference on Robotics and automation.

Acknowledgments

This project is developed under Department of Homeland Security (DHS) Science and Technology Assistance Agreement No. 2011-ST-108-000021 awarded by the U.S. Department of Homeland Security

Department of Homeland security

Figure E.4: Video Analytics Symposium

Bibliography

[1] M. Mori, K. F. MacDorman, and N. Kageki, "The uncanny valley [from the field]," *IEEE Robotics and Automation Magazine*, vol. 19, no. 2, pp. 98–100, 2012. ix, 16

[2] J. Zhang, Q. Zhang, and J. Hu, "Rgb color centroids segmentation (ccs) for face detection," *Graphics, Vision and Image Processing GVIP*, vol. 9, no. 2, pp. 1–9, 1 2009. ix, 21, 47, 48

[3] H. Sasaki, T. Fukuda, M. Satomi, and N. Kubota, "Growing neural gas for intelligent robot vision with range imaging camera," in *Mechatronics and Automation, 2009. ICMA 2009. International Conference on*, aug. 2009, pp. 3269 –3274. ix, xii, 67, 68, 69

[4] E. F. Nakamura, A. A. F. Loureiro, and A. C. Frery, "Information fusion for wireless sensor networks: Methods, models, and classifications," *ACM Comput. Surv.*, vol. 39, no. 3, Sep. 2007. x, 102, 103

[5] I. Laptev, "On space-time interest points," *International Journal of Computer Vision*, vol. 64, pp. 107–123, 2005, 10.1007/s11263-005-1838-7. [Online]. Available: http://dx.doi.org/10.1007/s11263-005-1838-7 3

[6] R. A. Harris, P. Graham, and T. S. Collett, "Visual cues for the retrieval of landmark memories by navigating wood ants," *Current biology*, vol. 17, no. 2, pp. 93–102, 2007. 3

[7] L. E. Parker, "Distributed intelligence: Overview of the field and its application in multi-robot systems," *Journal of Physical Agents (special issue on multi-robot systems)*, vol. 2, pp. 5–14, 2008. 4, 18

[8] G. Chen and A. Zakhor, "2d tree detection in large urban landscapes using aerial lidar data," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*, nov. 2009, pp. 1693 –1696. 4

[9] J. Yang, D. Schonfeld, and M. Mohamed, "Robust video stabilization based on particle filter tracking of projected camera motion," *IEEE Trans. Cir. and Sys. for Video Technol.*, vol. 19, no. 7, pp. 945–954, Jul. 2009. 4, 94

[10] I. Korpela, B. Dahlin, H. Schäfer, E. Bruun, F. Haapaniemi, J. Honkasalo, S. Ilvesniemi, V. Kuutti, M. Linkosalmi, J. Mustonen, Salo, O. Suomi, and H. Virtanen, "Single-tree forest inventory using lidar and aerial images for 3d treetop positioning, species recognition, height and crown width estimation," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 36, pp. 227–233, 2007. 4

[11] G. Andersen, "How to detect desert trees using corona images: Discovering historical ecological data," *Journal of Arid Environments*, vol. 65, no. 3, pp. 491 – 511, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140196305001928 4

[12] J. Juujrvi, J. Heikkonen, S. Brandt, and J. Lampinen, "Digital image based tree measurement for forest inventory," in *In Proceedings of SPIE The International Society for Optical Engineering: Intelligent Robots and Computer Vision XVII: Algorithms, Techniques, and Active Vision*, 1998, pp. 114–123. 4

[13] V. Jokelainen, "Recognition and localisation of young trees using machine perception," Master's thesis, Helsinki University of Technology, Espoo, Finland, 2010. 4

[14] W. Ali, F. Georgsson, and T. Hellstrom, "Visual tree detection for autonomous navigation in forest environment," in *Intelligent Vehicles Symposium, 2008 IEEE*, june 2008, pp. 560 –565. 4

[15] H. Niska, J.-P. Skon, P. Packalen, T. Tokola, M. Maltamo, and M. Kolehmainen, "Neural networks for the prediction of species-specific plot volumes using airborne laser scanning and aerial photographs," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 48, no. 3, pp. 1076 –1085, march 2010. 4

[16] P. Wang, W. Jiang, X. Li, S. Kang, and J. Xin, "Research for multi-sensor information fusion algorithm of search and rescue robot based on embedded control network," *Journal of Computers*, vol. 7, no. 5, pp. 1176–1183, May 2012. 4, 102

[17] Y. Satoh and K. Sakaue, "An omnidirectional stereo vision-based smart wheelchair," *EURASIP Journal on Image and Video Processing*, vol. 2007, no. 1, p. 087646, 2007. 5

[18] T. Razali, R. Zhu, K. Yoshinori, and K. Yoshinori, "Smart wheelchair navigation based on users gaze on destination," in *Advanced Intelligent Computing Theories and Applications*. Springer, 2010, pp. 387–394. 5

[19] C. King, M. V. Espina, R. Grech, R. Mullen, P. Remagnino, L. I. Iocchi, L. Marchetti, N. D. D. N. Monekosso, and M. Nicolescu, *Handbook on Soft Computing for Video Surveillance*. Taylor & Francis, 2011, ch. Multi-Robot and Multi-Camera Patrolling. 10, 102, 103, 104, 114

[20] M. V. Espina, R. Grech, D. de Jager, P. Remagnino, L. I. Iocchi, D. N. Monekosso, M. Nicolescu, and C. King, *Intelligent Paradigms in Security*, ser. INNOVATIONS IN DEFENCE SUPPORT SYSTEMS-3, 2011, ch. Multi-Robot Teams for Environmental Monitoring. 10, 12, 102, 103, 104, 114

[21] S. S. Ge and C. Fua, "Complete multi-robot coverage of unknown environments with minimum repeated coverage," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, april 2005, pp. 715 – 720. 10

[22] R. Grech, D. Monekosso, D. De Jager, and P. Remagnino, "A vision-based system for object identification and information retrieval in a smart home," in *Proceedings of the First international joint conference on Ambient intelligence*, ser. AmI'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 239–247. 11, 40, 41, 114, 115

[23] R. Grech, D. Monekosso, and P. Remagnino, "Building visual memories of video streams," *Electronics Letters*, vol. 48, no. 9, pp. 487 –488, 26 2012. 11, 45, 79, 115

[24] M. V. Espina, R. Grech, J. S. Cope, F. Felisberti, S. Mannan, D. Monekosso, and P. Remagnino, "Robot visual memory maps: compact encoding of saliency," *Robotics and Autonomous Systems*, (Submitted). 11, 12, 41, 45, 46, 50, 51, 52, 114, 115

[25] R. Grech, F. Flórez-Revuelta, D. Monekosso, and P. Remagnino, "Robot teams: Sharing visual memories," in *International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2012. 11, 46, 115

[26] S. A. Green, M. Billinghurst, X. Chen, and G. J. Chase, "Human-robot collaboration: A literature review and augmented reality approach in design," *International Journal of Advanced Robotic Systems*, vol. 5, no. 1, pp. 1–18, 2008. 13

[27] L. E. Parker, "Interview with lynne e. parker," *International Journal of Advanced Robotic Systems*, vol. 2, no. 2, June 2004. [Online]. Available: http://www.ars-journal.com/ars/Interview/Interview+with+Lynne+E.htm 14

[28] M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Galvez-Lopez, K. Haussermann, R. Janssen, J. Montiel, A. Perzylo, B. Schiessle, M. Tenorth, O. Zweigle, and R. van de Molengraft, "Roboearth," *Robotics Automation Magazine, IEEE*, vol. 18, no. 2, pp. 69 –82, june 2011. 14, 36, 95, 101, 102

[29] X. Zhao, Q. Luo, and B. Han, "Survey on robot multi-sensor information fusion technology," in *Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on*, june 2008, pp. 5019 –5023. 15, 102

[30] S. M. Singer and D. L. Akin, "A survey of quantitative team performance metrics for human-robot collaboration," *41st International Conference on Environmental Systems*, 17-21 July 2011. 15, 16

[31] M. A. Goodrich and A. C. Schultz, "Human-robot interaction: a survey," *Found. Trends Hum.-Comput. Interact.*, vol. 1, no. 3, pp. 203–275, Jan. 2007. [Online]. Available: http://dx.doi.org/10.1561/1100000005 15

[32] R. Murphy, "Human-robot interaction in rescue robotics," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 34, no. 2, pp. 138 –153, may 2004. 16

[33] R. Nourjou, M. Hatayama, and H. Tatano, "Introduction to spatially distributed intelligent assistant agents for coordination of human-agent teams' actions," in *Safety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium on*, nov. 2011, pp. 251 –258. 16

[34] J. Seyama and R. S. Nagayama, "The uncanny valley: Effect of realism on the impression of artificial human faces," *Presence: Teleoperators and Virtual Environments*, vol. 16, no. 4, pp. 337–351, 2007. 17

[35] J. M. Beer, A. Prakash, T. L. Mitzner, and W. A. Rogers, "Understanding robot acceptance - smartech," Georgia Institute of Technology. Human Factors and Aging Laboratory, Tech. Rep. HFA-TR-1103, 2011. 17

[36] P. Levi and S. Kernbach, "Symbiotic Multi-Robot Organisms: Reliability, Adaptability, Evolution," 2010. 17, 32, 38

[37] E. Şahin, "Swarm robotics: From sources of inspiration to domains of application," in *Swarm Robotics*, ser. Lecture Notes in Computer Science, E. Şahin and W. M. Spears, Eds., vol. 3342. Springer, 2005, pp. 10–20. 17

[38] P. Muniganti and A. O. Pujol, *A Survey on Mathematical models of Swarm Robotics*, 2010. [Online]. Available: http://www.jopha.net/waf/index.php/waf/waf10/paper/viewPDFInterstitial/67/72 17

[39] T. Fukuda, S. Nakagawa, Y. Kawauchi, and M. Buss, "Self organizing robots based on cell structures - ckbot," in *Intelligent Robots, 1988., IEEE International Workshop on*, oct-2 nov 1988, pp. 145 –150. 17

[40] Y. Davidor, *Genetic Algorithms and Robotics: A Heuristic Strategy for Optimization*. World Scientific, 1991. 17

[41] L. E. Parker, "Current research in multirobot systems," *Journal of Artificial Life And Robotics*, vol. 7, no. 1/2, pp. 1–5, 2003. 18, 19, 30, 131, 134

[42] R. Rocha, J. Dias, and A. Carvalho, "Cooperative multi-robot systems a study of vision-based 3-d mapping using information theory," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, april 2005, pp. 384 – 389. 18, 24, 37, 131, 134, 141, 142, 143

[43] P. Bechon and J.-J. E. Slotine, "Synchronization and quorum sensing in a swarm of humanoid robots," *CoRR*, vol. abs/1205.2952, 2012. 19

[44] A. Shafi and C. Holton, "Market opportunity for machine vision in service robots," *Vision Systems Design*, 2011. 19, 119

[45] B. G. Batchelor, "Coming to terms with machine vision and computer vision: they're not the same!" *Advanced Imaging, Cygnus Business Media Inc.*, pp. 22–26, 1999. 19

[46] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media Inc., 2008. [Online]. Available: http://oreilly.com/catalog/9780596516130 20, 21, 22, 29, 57, 58, 126, 127

[47] F. Remondino and C. Fraser, "Digital camera calibration methods: considerations and comparisons," in *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences,*, Isprs, Ed., vol. Vol. XXXVI, no. part 5, Dresden, Germany, 2006. 20

[48] S. Bhattacharyya, "A brief survey of color image preprocessing and segmentation techniques," *Journal of Pattern Recognition Research*, vol. 6, no. 1, pp. 120–129, 2011. 20, 121

[49] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision*, 2002. 21

[50] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vision*, vol. 57, pp. 137–154, May 2004. 21

[51] S. Jones, C. Andresen, and J. Crowley, "Appearance based process for visual navigation," in *Intelligent Robots and Systems, 1997. IROS '97., Proceedings of the 1997 IEEE/RSJ International Conference on*, vol. 2, sep 1997, pp. 551 –557 vol.2. 22, 130

[52] Z. Chen, J. Samarabandu, and R. Rodrigo, "Recent advances in simultaneous localization and map-building using computer vision," *Advanced Robotics*, vol. 21, no. 3, pp. 233–265, 2007. 22, 23, 26, 28, 30

[53] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004. 23

[54] Z. Zhang and T. Kanade, "Determining the epipolar geometry and its uncertainty: A review," *International Journal of Computer Vision*, vol. 27, pp. 161–195, 1998. 23

[55] O. Faugeras, Q.-T. Luong, and T. Papadopoulou, *The Geometry of Multiple Images: The Laws That Govern The Formation of Images of A Scene and Some of Their Applications*. Cambridge, MA, USA: MIT Press, 2001. 23

[56] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry, *An Invitation to 3-D Vision: From Images to Geometric Models*. SpringerVerlag, 2003. 23

[57] Y. Lu, J. Z. Zhang, Q. M. J. Wu, and Z. nian Li, "A survey of motion-parallax-based 3d reconstruction algorithms," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 34, pp. 532–548, 2004. 23

[58] A. Fusiello, "Uncalibrated Euclidean reconstruction: A review," *Image and Vision Computing*, vol. 18, no. 6–7, pp. 555–563, May 2000. 23

[59] U. Nehmzow, *Mobile Robotics: A Practical Introduction*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2003. 23

[60] A. Gil, ı. Reinoso, M. Ballesta, and M. Juliá, "Multi-robot visual slam using a rao-blackwellized particle filter," *Robot. Auton. Syst.*, vol. 58, no. 1, pp. 68–80, 2010. 24, 25, 26, 27, 28, 125

[61] S. Thrun, "Robotic mapping: A survey," in *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002. 24

[62] H. Durrant-Whyte and T. Bailey, "Simultaneous localisation and mapping (slam): Part i the essential algorithms," *IEEE Robotics and Automation Magazine*, vol. 13(2), pp. 99–108, 2006. 24

[63] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (slam): Part ii: State of the art," *IEEE Robotics and Automation Magazine*, vol. 13(3), pp. 108–117, 2006. 24

[64] A. Gil, O. Mozos, M. Ballesta, and O. Reinoso, "A comparative evaluation of interest point detectors and local descriptors for visual slam," *Machine Vision and Applications*, vol. 21, pp. 905–920, 2010. 24, 125

[65] M. Mendes, A. P. Coimbra, and M. M. Crisóstomo, "Topological mapping using vision and a sparse distributed memory," in *Electrical Engineering and Applied Computing*, ser. Lecture Notes in Electrical Engineering, S.-I. Ao and L. Gelman, Eds. Springer Netherlands, 2011, vol. 90, pp. 273–284. 24, 30, 64, 81, 130

[66] S. Kirstein, H. Wersing, and E. Körner, "A biologically motivated visual memory architecture for online learning of objects," *Neural Networks*, vol. 21, pp. 65–77, January 2008. 25, 66, 79

[67] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 237–267, 2002. 25

[68] V. Högman, "Building a 3d map from rgb-d sensors," Master's thesis, Computer Vision and Active Perception Laboratory, Royal Institute of Technology (KTH), Stockholm, Sweden, February 2012. 26, 27, 28, 124, 126

[69] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," pp. 167–193, 1990. 26

[70] M. Calonder, "Ekf slam vs. fastslam a comparison," Tech. Rep., 2006. 26

[71] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani, and H. Durrant-Whyte, "Simultaneous localization and mapping with sparse extended information filters," *The International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 693–716, August 2004. 26

[72] N. E. Özkucur and H. L. Akin, "Cooperative multi-robot map merging using fast-slam," in *RoboCup*, 2009, pp. 449–460. 27

[73] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, 2007. 28, 122

[74] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proceedings of the AAAI National Conference on Artificial Intelligence*. Edmonton, Canada: AAAI, 2002. 28

[75] M. Montemerlo, S. Thrun, D. Roller, and B. Wegbreit, "Fastslam 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *IJCAI'03: Proceedings of the 18th international joint conference on Artificial intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003, pp. 1151–1156. 28

[76] M. Milford, R. Schulz, D. Prasser, G. Wyeth, and J. Wiles, "Learning spatial concepts from ratslam representations," *Robot. Auton. Syst.*, vol. 55, no. 5, pp. 403–410, 2007. 28

[77] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007. 28

[78] R. Castle, G. Klein, and D. W. Murray, "Video-rate localization in multiple maps for wearable augmented reality," in *ISWC '08: Proceedings of the 2008 12th IEEE International Symposium on Wearable Computers*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 15–22. 28

[79] E. Eade and T. Drummond, "Edge landmarks in monocular slam," *Image Vision Comput.*, vol. 27, no. 5, pp. 588–596, 2009. 28

[80] E. Eade and T. Drummond, "Scalable monocular slam," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1*, ser. CVPR '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 469–476. 28

[81] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainly," *Int. J. Rob. Res.*, vol. 5, no. 4, pp. 56–68, 1987. 28

[82] J. Neira and J. Tardós, *IEEE Transactions on Robotics and Automation*, vol. Vol. 17, no. No. 6, pp. pp. 890 – 897, December 2001. 28

[83] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981. 28

[84] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments," *I. J. Robotic Res.*, vol. 31, no. 5, pp. 647–663, 2012. 28

[85] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," *Autonomous Agents and Multi-Agent Systems*, vol. 11, pp. 387–434, 2005, 10.1007/s10458-005-2631-2. 30, 34, 38, 39, 132, 133

[86] J. Wolf, W. Burgard, and H. Burkhardt, "Robust vision-based localization for mobile robots using an image retrieval system based on invariant features," in *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 2002. 30, 129

[87] A. Remazeilles, F. Chaumette, and P. Gros, "Robot motion control from a visual memory," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 5, april-1 may 2004, pp. 4695 – 4700 Vol.5. 30

[88] G. Blanc, Y. Mezouar, and P. Martinet, "Indoor navigation of a wheeled mobile robot along visual routes," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, april 2005, pp. 3354 – 3359. 30

[89] J. Wolf, W. Burgard, and H. Burkhardt, "Robust vision-based localization by combining an image-retrieval system with monte carlo localization," *Robotics, IEEE Transactions on*, vol. 21, no. 2, pp. 208 – 216, april 2005. 30

[90] A. Remazeilles, F. Chaumette, and P. Gros, "3d navigation based on a visual memory," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, may 2006, pp. 2719 –2725. 30

[91] A. C. Murillo, C. Sagüés, J. J. Guerrero, T. Goedemé, T. Tuytelaars, and L. Van Gool, "From omnidirectional images to hierarchical localization," *Robot. Auton. Syst.*, vol. 55, no. 5, pp. 372–382, 2007. 30, 129

[92] A. Cherubini, M. Colafrancesco, G. Oriolo, L. Freda, and F. Chaumette, "Comparing appearance-based controllers for nonholonomic navigation from a visual memory," in *ICRA 2009 Workshop on safe navigation in open and dynamic environments: application to autonomous vehicles*, Kobe, Japan, May 2009. 30

[93] Y. Cao, A. Fukunaga, A. Kahng, and F. Meng, "Cooperative mobile robotics: antecedents and directions," in *Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, vol. 1, aug 1995, pp. 226 –234 vol.1. 31, 34, 38, 133

[94] Y. Yan and T. Zhenmin, "Control architecture for autonomous multi-robot system: Survey and analysis," in *Intelligent Computation Technology and Automation, 2009. ICICTA '09. Second International Conference on*, vol. 4, oct. 2009, pp. 376 –379. 31, 36, 37, 38

[95] G. Mckee and B. Varghese, "Robot teams and robot team players," *Multi-Robot Systems, Trends and Development,Toshiyuki Yasuda (Ed.)*, January 2011. 31, 32, 34, 35, 36, 37, 131, 135

[96] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated multi-robot exploration," *Robotics, IEEE Transactions on*, vol. 21, no. 3, pp. 376 – 386, june 2005. 31, 34, 35

[97] L. Iocchi, D. Nardi, and M. Salerno, "Reactivity and deliberation: a survey on multi-robot systems," *Lecture Notes in Computer Science*, vol. 2103, pp. 9–32, 2001. [Online]. Available: http://www.springerlink.com/index/qwlwbahk0nx8hun0.pdf 32, 33, 34, 36, 37, 133, 134, 135

[98] B. P. Gerkey and M. J. Mataric, "A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, Sep. 2004. 32, 35, 36, 134

[99] N. Agmon, "On events in multi-robot patrol in adversarial environments," in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 2 - Volume 2*, ser. AAMAS '10. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2010, pp. 591–598. [Online]. Available: http://dl.acm.org/citation.cfm?id=1838178.1838184 33

[100] D. Vallejo, P. Remagnino, D. N. Monekosso, L. Jiménez, and C. González, "A multi-agent architecture for multi-robot surveillance," in *Proceedings of the 1st International Conference on Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems*, ser. ICCCI '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 266–278. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-04441-0_23 33

[101] E. Folgado, M. Rincón, J. R. Álvarez, and J. Mira, "A multi-robot surveillance system simulated in gazebo," in *Proceedings of the 2nd international work-conference on Nature Inspired Problem-Solving Methods in Knowledge Engineering: Interplay Between Natural and Artificial Computation, Part II*, ser. IWINAC '07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 202–211. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-73055-2_22 33

[102] L. Parker, "Cooperative motion control for multi-target observation," in *Intelligent Robots and Systems, 1997. IROS '97., Proceedings of the 1997 IEEE/RSJ International Conference on*, vol. 3, sep 1997, pp. 1591 -1597 vol.3. 33

[103] W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun, "Collaborative multi-robot exploration," in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 1, 2000, pp. 476 -481 vol.1. 33

[104] F. Zhang, W. Chen, and Y. Xi, "Improving collaboration through fusion of bid information for market-based multi-robot exploration," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, april 2005, pp. 1157 - 1162. 33

[105] Y. Feng, Z. Zhu, and J. Xiao, "Heterogeneous multi-robot localization in unknown 3d space," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, oct. 2006, pp. 4533 -4538. 33

[106] A. Renzaglia and A. Martinelli, "Potential field based approach for coordinate exploration with a multi-robot team," in *Safety Security and Rescue Robotics (SSRR), 2010 IEEE International Workshop on*, july 2010, pp. 1 -6. 33

[107] G. G. Rigatos, "Multi-robot motion planning using swarm intelligence," *International Journal of Advanced Robotic Systems*, vol. 5, no. 2, p. 139, June 2008. 33

[108] Y. Ktiri, T. Yoshikai, and M. Inaba, "Multi-robot exploration framework using robot vision and laser range data," in *System Integration (SII), 2011 IEEE/SICE International Symposium on*, dec. 2011, pp. 1251 -1256. 33

[109] L. Zheng, "Multi-robot cooperative surveillance in unknown environments," Ph.D. dissertation, Department of Electrical & Computer Engineering, National University of Singapore, 2006. 33

[110] F. Almeida, N. Lau, and L. P. Reis, "A survey on coordination methodologies for simulated robotic soccer teams," in *MALLOW*, ser. CEUR Workshop Proceedings, O. Boissier, A. E. Fallah-Seghrouchni, S. Hassas, and N. Maudet, Eds., vol. 627. CEUR-WS.org, 2010. 33, 101, 134

[111] J. Kiener and O. von Stryk, "Towards cooperation of heterogeneous, autonomous robots: A case study of humanoid and wheeled robots," *Robot. Auton. Syst.*, vol. 58, no. 7, pp. 921-929, Jul. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.robot.2010.03.013 33

[112] E. Pagello, A. D'Angelo, and E. Menegatti, "Cooperation issues and distributed sensing for multirobot systems," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1370 –1383, july 2006. 33

[113] L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 38, no. 2, pp. 156 –172, march 2008. 34, 38, 39

[114] A. R. Mosteo and L. Montano, "A survey of multi-robot task allocation." 35, 37, 131, 132

[115] S. Nouyan, R. Gross, M. Bonani, F. Mondada, and M. Dorigo, "Teamwork in self-organized robot colonies," *Evolutionary Computation, IEEE Transactions on*, vol. 13, no. 4, pp. 695 –711, aug. 2009. 35

[116] J. Bradshaw, P. Feltovich, M. Johnson, L. Bunch, M. Breedy, T. Eskridge, H. Jung, J. Lott, and A. Uszok, "Coordination in human-agent-robot teamwork," in *Collaborative Technologies and Systems, 2008. CTS 2008. International Symposium on*, may 2008, pp. 467 –476. 35

[117] G. Hoffman and C. Breazeal, "Effects of anticipatory action on human-robot teamwork efficiency, fluency, and perception of team," in *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, ser. HRI '07. New York, NY, USA: ACM, 2007, pp. 1–8. [Online]. Available: http://doi.acm.org/10.1145/1228716.1228718 35

[118] S.-L. Kim, W. Burgard, and D. Kim, "Wireless communications in networked robotics [guest editorial]," *Wireless Communications, IEEE*, vol. 16, no. 1, pp. 4 –5, february 2009. 36, 131, 132

[119] M. J. Matarić, G. S. Sukhatme, and E. Østergaard, "Multi-robot task allocation in uncertain environments," *Autonomous Robots*, vol. 14, no. 2, pp. 255–263, 2003. [Online]. Available: http://cres.usc.edu/cgi-bin/print_pub_details.pl?pubid=228 36

[120] L. Vig and J. A. Adams, "Market-based multi-robot coalition formation," pp. 227–236, 2006. 36

[121] G. A. Kaminka and I. Frenkel, "Flexible teamwork in behavior-based robots," in *Proceedings of the 20th national conference on Artificial intelligence - Volume 1*. AAAI Press, 2005, pp. 108–113. [Online]. Available: http://portal.acm.org/citation.cfm?id=1619352 37

[122] G. Dudek and M. Jenkin, *Computational Principles of Mobile Robotics*, 2nd ed. New York, NY, USA: Cambridge University Press, 2010. 37

[123] E. Hernandez-Martinez and J. Albino, "Hybrid architecture of multi-robot systems based on formation control and som neural networks," in *Control Applications (CCA), 2011 IEEE International Conference on*, sept. 2011, pp. 941–946. 37

[124] C. Versino and L. M. Gambardella, "Learning real team solutions," in *Selected papers from the Workshop on Distributed Artificial Intelligence Meets Machine Learning, Learning in Multi-Agent Environments*, ser. ECAI '96. London, UK, UK: Springer-Verlag, 1997, pp. 40–61. 38

[125] T. Balch, "Learning roles: Behavioral diversity in robot teams." AAAI, 1997, pp. 7–12. 38

[126] T. Balch, "Social entropy: a new metric for learning multi-robot teams," in *In Proc. 10th International FLAIRS Conference (FLAIRS-97)*, 1997, pp. 272–277. 38

[127] T. Balch, "Behavioral diversity in learning robot teams," Ph.D. dissertation, Atlanta, GA, USA, 1998. 38

[128] T. Balch, "Hierarchic social entropy: An information theoretic measure of robot group diversity," *Auton. Robots*, vol. 8, no. 3, pp. 209–238, Jun. 2000. 38

[129] L. E. Parker, C. Touzet, and D. Jung, "Learning and adaptation in multi-robot teams," in *Proc. 18th Symp. on Energy Engineering Sciences*, 2000, pp. 177–185. 38

[130] L. E. Parker, C. Touzet, and F. Fernandez, *Robot Teams: From Diversity to Polymorphism*. A K Peters, 2002, ch. Techniques for Learning in Multi-Robot Teams, pp. 191–236. 38

[131] E. Yang and D. Gu. (2004) Multiagent Reinforcement Learning for Multi-Robot Systems: A Survey. 38, 126

[132] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254–1259, Nov. 1998. 41, 43, 44, 45

[133] L. Itti and C. Koch, "Computational modelling of visual attention." *Nature reviews. Neuroscience*, vol. 2, no. 3, pp. 194–203, Mar. 2001. 41

[134] J. K. Tsotsos, A. J. Rodrguez-Snchez, A. L. Rothenstein, and E. Simine, "Different binding strategies for the different stages of visual recognition." in *BVAI*, ser. Lecture Notes in Computer Science, F. Mele, G. Ramella, S. Santillo, and F. Ventriglia, Eds., vol. 4729. Springer, 2007, pp. 150–160. 41

[135] N. D. B. Bruce and J. K. Tsotsos, "Saliency, attention, and visual search: An information theoretic approach," *Journal of Vision*, vol. 9, no. 3, pp. 1–24, 2009. 41

[136] Z. Li and L. Itti, "Saliency and gist features for target detection in satellite images." *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 20, no. 7, pp. 2017–2029, Jul. 2011. 41

[137] R. Achanta, S. S. Hemami, F. J. Estrada, and S. Susstrunk, "Frequency-tuned salient region detection," in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*. IEEE, 2009, pp. 1597–1604. 41, 43

[138] J. Huang, X. Yang, R. Zhang, F. Lu, and X. Fang, "Integrating visual saliency and consistency for re-ranking image search results," in *Proceedings of the International Conference on Image Processing*, 2010, pp. 3173–3176. 41

[139] C.-K. Chang, C. Siagian, and L. Itti, "Mobile robot vision navigation amp; localization using gist and saliency," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, oct. 2010, pp. 4147–4154. 41, 43

[140] A. V. Nefian. Georiga tech face database. [Online]. Available: http://www.anefian.com/research/face_reco.htm 41

[141] M. Corbetta and G. L. Shulman, "Control of goal-directed and stimulus driven attention in the brain," *Nature Reviews Neuroscience*, vol. 3 (3), pp. 201–215, 2002. 42

[142] L. Pessoa, S. Kastner, and L. G. Ungerleider, "Neuroimaging studies of attention: from modulation of sensory processing to top-down control." *The Journal of neuroscience : the official journal of the Society for Neuroscience*, vol. 23, no. 10, pp. 3990–3998, May 2003. 42

[143] S. Goferman, L. Zelnik-Manor, and A. Tal, "Context-Aware Saliency Detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 43

[144] Z. Liu, Y. Xue, H. Yan, and Z. Zhang, "Efficient saliency detection based on gaussian models," *Image Processing, IET*, vol. 5, no. 2, pp. 122–131, march 2011. 43

[145] F. Urban, B. Follet, C. Chamaret, O. L. Meur, and T. Baccino, "Medium spatial frequencies, a strong predictor of salience," *Cognitive Computation*, pp. 37–47, 2011. 43

[146] W. Einhauser, M. Spain, and P. Perona, "Objects predict fixations better than early saliency," *Journal of Vision*, vol. 8, no. 14, pp. 1–26, 2008. 43

[147] D. He, Y. Zhang, and H. Song, "A novel saliency map extraction method based on improved itti's model," in *Computer and Communication Technologies in Agriculture Engineering (CCTAE), 2010 International Conference On*, vol. 3, june 2010, pp. 323–327. 43

[148] J. Bonaiuto and L. Itti, "Combining attention and recognition for rapid scene analysis," in *Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, june 2005, p. 90. 43

[149] M. Milanova, S. Rubin, R. Kountchev, V. Todorov, and R. Kountcheva, "Combined visual attention model for video sequences," in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, dec. 2008, pp. 1–4. 43

[150] T. Yubing, F. A. Cheikh, F. F. E. Guraya, H. Konik, and A. Trmeau, "A spatiotemporal saliency model for video surveillance," *Cognitive Computation*, pp. 241–263, 2011. 43

[151] T. Lindeberg, "Feature detection with automatic scale selection," *International Journal of Computer Vision*, vol. 30, no. 2, pp. 79–116, Nov. 1998. [Online]. Available: http://dx.doi.org/10.1023/A:1008045108935 43

[152] C. Ngau, L.-M. Ang, and K. P. Seng, "Bottom-up visual saliency map using wavelet transform domain," in *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, vol. 1, july 2010, pp. 692–695. 43

[153] Z. Su and S. Takahashi, "Real-time enhancement of image and video saliency using semantic depth of field," in *VISAPP (2)'10*, 2010, pp. 370–375. 43, 45

[154] Y. Lin, Y. Y. Tang, B. Fang, Z. Shang, Y. Huang, and S. Wang, "A visual-attention model using earth mover's distance based saliency measurement and

nonlinear feature combination," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), In press*, 2012. 43

[155] N. Butko, L. Zhang, G. Cottrell, and J. Movellan, "Visual saliency model for robot cameras," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, may 2008, pp. 2398 –2403. 43

[156] A. Borji and L. Itti, "State-of-the-art in visual attention modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence, In press*, 2012. 43, 44

[157] A. Zeileis, K. Hornik, and P. Murrell, "Escaping rgbland: Selecting colors for statistical graphics," *Computational Statistics & Data Analysis*, vol. 53, no. 9, pp. 3259 – 3270, 2009. 50

[158] M. Merler, C. Galleguillos, and S. Belongie, "Recognizing groceries in situ using in vitro training data." in *CVPR*. IEEE Computer Society, 2007. 54

[159] P. Narasimhan, R. Gandhi, and D. Rossi, "Smartphone-based assistive technologies for the blind," in *Proceedings of the 2009 international conference on Compilers, architecture, and synthesis for embedded systems*, ser. CASES '09. New York, NY, USA: ACM, 2009, pp. 223–232. [Online]. Available: http://doi.acm.org/10.1145/1629395.1629427 54

[160] J. Hallberg, C. Nugent, R. Davies, and M. Donnelly, "Localisation of forgotten items using rfid technology," in *Information Technology and Applications in Biomedicine, 2009. ITAB 2009. 9th International Conference on*, nov. 2009, pp. 1 –4. 54

[161] V. Kulyukin, C. Gharpure, P. Sute, N. De Graw, and J. Nicholson, "A robotic wayfinding system for the visually impaired," in *Proceedings of the 16th conference on Innovative applications of artifical intelligence*, ser. IAAI'04. AAAI Press, 2004, pp. 864–869. [Online]. Available: http://dl.acm.org/citation.cfm?id=1597321.1597337 54

[162] V. Kulyukin, C. Gharpure, and J. Nicholson, "Robocart: toward robot-assisted navigation of grocery stores by the visually impaired," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, aug. 2005, pp. 2845 – 2850. 54

[163] J. Coughlan and R. Manduchi, "Color targets: Fiducials to help visually impaired people find their way by camera phone," *EURASIP Journal on Image and Video Processing*, vol. 2007, no. 2, p. 10–10, 2007. [Online]. Available: http://www.hindawi.com/journals/ivp/2007/096357.abs.html 54

[164] J. M. Coughlan and R. Manduchi, "Functional assessment of a camera phone-based wayfinding system operated by blind and visually impaired users," *International Journal on Artificial Intelligence Tools*, vol. 18, no. 3, pp. 379–397, 2009. 54

[165] G. Saponaro and A. Bernardino, "Bootstrapping Visual Memories for a Humanoid Robot," in *15th Portuguese Conference on Pattern Recognition (RecPad 2009)*, Aveiro, Portugal, October 2009. 63, 66, 78

[166] A. D. Baddeley, *Human memory : theory and practice*. Boston: Allyn and Bacon, 1990. 64

[167] E. Matin, "Saccadic suppression: a review and an analysis," *Psychological Bulletin*, vol. 81, no. 12, pp. 899–917, 1974. 65

[168] A. Hollingworth, "Memory for object position in natural scenes," *Visual Cognition*, vol. 12, pp. 1003–1016, 2005. 65

[169] T. Konkle, T. F. Brady, G. A. Alvarez, and A. Oliva, "Scene memory is more detailed than you think: The role of categories in visual long-term memory," *Psychological Science, In Press*, 2010. 65

[170] T. Konkle, T. F. Brady, G. A. Alvarez, and O. Aude, "Conceptual distinctiveness supports detailed visual long-term memory," *Journal of Experimental Psychology*, vol. General, 139 (3), pp. 558–78s, 2010. 65

[171] R. Q. Quiroga, I. Fried, and C. Koch, "Brain cells for grandmother," *Neuroscience, Scientific American*, vol. 308, pp. 30 – 35, February 2013. 65

[172] X. Cao and P. N. Suganthan, "Neural network based temporal video segmentation," *Int. J. Neural Syst.*, vol. 12, no. 3-4, pp. 263–269, 2002. 66, 75, 77, 78

[173] D. Kit, B. Sullivan, and D. Ballard, "Novelty detection using growing neural gas for visuo-spatial memory," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, sept. 2011, pp. 1194 –1200. 66, 130

[174] M. Green. Eyewitness memory is unreliable,(last accessed on 10/03/2012). [Online]. Available: http://www.visualexpert.com/Resources/eyewitnessmemory.html 67, 104

[175] C. Nutbrown, *Threads of Thinking: Schemas and Young Children's Learning*. SAGE Publications, 2011. 67

[176] S. Vosniadou, *How Children Learn,*. Educational Practices, Series, 7, The International Academy of Education (IAE) and the International Bureau of Education (UNESCO), 2001. 67

[177] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464 –1480, Sep. 1990. 67, 128

[178] K.-L. Du, "Clustering: A neural network approach," *Neural Networks*, vol. 23, no. 1, pp. 89–107, January 2010. 67, 127

[179] M. M. V. Hulle, "Self-organising maps," vol. Chapter 1. [Online]. Available: www.pspc.dibe.unige.it/~drivsco/Papers/VanHulle_Springer.pdf 67

[180] T. Furukawa, "Som of soms," *Neural Networks*, vol. 22, no. 4, pp. 463 – 478, 2009. 67

[181] B. Fritzke, "A growing neural gas network learns topologies," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. S. Touretzky, and T. K. Leen, Eds. Cambridge MA: MIT Press, 1995, pp. 625–632. 67, 69, 128, 146

[182] Q. Le, G. Anagnostopoulos, M. Georgiopoulos, and K. Ports, "An experimental comparison of semi-supervised artmap architectures, gcs and gng classifiers," in *Neural Networks, 2005. IJCNN '05. Proceedings. 2005 IEEE International Joint Conference on*, vol. 5, july-4 aug. 2005, pp. 3121 – 3126 vol. 5. 68, 69

[183] F. Florez, J. Garcia, J. Garcia, and A. Hernández, "Representing 2d objects. comparison of several self-organizing networks," *3th WSES Conference on Neural Networks ancl Applications, Interlaken*, pp. 69–72, 2002. 68, 70

[184] D. Heinke and F. Hamker, "Comparing neural networks: a benchmark on growing neural gas, growing cell structures, and fuzzy artmap," *Neural Networks, IEEE Transactions on*, vol. 9, no. 6, pp. 1279 –1291, nov 1998. 68

[185] J. Holmström, "Growing Neural Gas Experiments with GNG, GNG with Utility and Supervised GNG," Master's thesis, Uppsala University. 68

[186] J. García-Rodríguez, F. Flórez-Revuelta, and J. M. García-Chamizo, "Hybrid gng architecture learns features in images," in *Proceedings of the 3rd international workshop on Hybrid Artificial Intelligence Systems*, ser. HAIS '08. Berlin, Heidelberg: Springer-Verlag, *2008, pp.* 451–457. 70

[187] G. Donatti and R. Wurtz, "Using growing neural gas networks to represent visual object knowledge," in *Tools with Artificial Intelligence, 2009. ICTAI '09. 21st International Conference on*, nov. 2009, pp. 54 –58. 70

[188] R. Laganière, *OpenCV 2 Computer Vision Application Programming Cookbook*. Packt Publishing, May 2011. 72

[189] M. S. Ahmed, R. Saatchi, and F. Caparrelli, "Vision based object recognition and localisation by a wireless connected distributed robotic systems," *Electronic Letters on Computer Vision and Image Analysis*, vol. 11(1), pp. 54–67, 2012. 72, 126

[190] L. Wang, Y. Zhang, and J. Feng, "On the euclidean distance of images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 8, pp. 1334 –1339, aug. 2005. 73, 74, 123

[191] B. Sun and J. Feng, "A fast algorithm for image euclidean distance," in *Pattern Recognition, 2008. CCPR '08. Chinese Conference on*, oct. 2008, pp. 1 –5. 73, 74

[192] P. Chiu, A. Girgensohn, W. Polak, E. Rieffel, and L. Wilcox, "A genetic algorithm for video segmentation and summarization," in *IEEE International Conference on Multimedia and Expo*. New York, NY: IEEE, 2000. 75

[193] S. Porter, "Video segmentation and indexing using motion estimation," Ph.D. dissertation, University of Bristol, February 2004. 75

[194] X. Song and G. Fan, "Selecting salient frames for spatiotemporal video modeling and segmentation," *Image Processing, IEEE Transactions on*, vol. 16, no. 12, pp. 3035 –3046, dec. 2007. 75

[195] M. Chatzigiorgaki and A. Skodras, "Real-time keyframe extraction towards video content identification," in *Digital Signal Processing, 2009 16th International Conference on*, july 2009, pp. 1 –6. 75, 77, 82

[196] J. Lezama, K. Alahari, J. Sivic, and I. Laptev, "Track to the future: Spatio-temporal video segmentation with long-range motion cues," pp. 3369 –3376, june 2011. 75

[197] Y. Ding and G. Fan, "Sports video mining via multichannel segmental hidden markov models," *Multimedia, IEEE Transactions on*, vol. 11, no. 7, pp. 1301 –1309, nov. 2009. 75

[198] V. V and N. R, "Recent trends and research issues in video association mining," *CoRR*, vol. abs/1112.2040, 2011. 75

[199] I. Koprinska and S. Carrato, "Temporal video segmentation: A survey," *Signal Processing: Image Communication*, vol. 16, no. 5, pp. 477 – 500, 2001. 75

[200] S. Lefèvre, J. Holler, and N. Vincent, "A review of real-time segmentation of uncompressed video sequences for content-based search and retrieval," *Real-Time Imaging*, vol. 9, pp. 73–98, February 2003. 75

[201] X. Zhu, X. Wu, A. Elmagarmid, Z. Feng, and L. Wu, "Video data mining: semantic indexing and event detection from the association perspective," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, no. 5, pp. 665 – 677, May 2005. 75

[202] W. Gao, Y. Tian, T. Huang, and Q. Yang, "Vlogging: A survey of videoblogging technology on the web," *ACM Comput. Surv.*, vol. 42, pp. 15:1–15:57, June 2010. 75, 78

[203] K. N. Ngan and H. Li, *Video Segmentation and Its Applications*, 1st ed. New York, NY : Springer, 2011. 75, 77, 78, 88

[204] C. Siagian and L. Itti, "Storing and recalling information for vision localization." in *ICRA*. IEEE, 2008, pp. 1848–1855. 78

[205] B. Fan and J. Pu, *Multi-Robot Information Fusion and Coordination Based on Agent*. Multi-Robot Systems, Trends and Development, 2011, ch. 18. 95

[206] L. Iocchi, D. Monekosso, D. Nardi, M. Nicolescu, P. Remagnino, and M. Valera Espina, "Smart monitoring of complex public scenes," in *2011 AAAI Fall Symposium Series*, 2011. 96

[207] J. Kramer and M. Scheutz, "Development environments for autonomous mobile robots: A survey," *Autonomous Robots*, vol. 22, p. 132, 2007. 97, 135, 136

[208] N. Mohamed, J. Al-Jaroodi, and I. Jawhar, "Middleware for robotics: A survey," in *Robotics, Automation and Mechatronics, 2008 IEEE Conference on*, sept. 2008, pp. 736 –742. 97, 98, 99, 136

[209] I. Lütkebohle, R. Philippsen, V. Pradeep, E. Marder-Eppstein, and S. Wachsmuth, "Generic middleware support for coordinating robot software components : The task-state-pattern," *Journal of Software Engineering for Robotics*, vol. 2, no. 1, pp. 20–39, 2011. 98, 99, 135, 136

[210] M. Valera and S. Velastin, "An approach for designing a real-time intelligent distributed surveillance system," in *Intelligence Distributed Surveillance Systems, IEE Symposium on (Ref. No. 2003/10062)*, feb. 2003, pp. 6/1 – 6/5. 99

[211] F. Simoes, M. Almeida, M. Pinheiro, R. dos Anjos, A. dos Santos, R. Roberto, V. Teichrieb, C. Suetsugo, and A. Pelinson, "Challenges in 3d reconstruction

from images for difficult large-scale objects: A study on the modeling of electrical substations," in *Virtual and Augmented Reality (SVR), 2012 14th Symposium on*, may 2012, pp. 74 –83. 100

[212] C. Kemp, A. Edsinger, and E. Torres-Jara, "Challenges for robot manipulation in human environments [grand challenges of robotics]," *Robotics Automation Magazine, IEEE*, vol. 14, no. 1, pp. 20 –29, march 2007. 100

[213] L. Marchetti, G. Grisetti, and L. Iocchi, "Robocup *2006: Robot soccer world* cup x," G. Lakemeyer, E. Sklar, D. G. Sorrenti, and T. Takahashi, Eds. Berlin, Heidelberg: Springer-Verlag, 2007, ch. A Comparative Analysis of Particle Filter Based Localization Methods, pp. 442–449. 100

[214] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *Trans. Rob.*, vol. 23, no. 1, pp. 34–46, Feb. 2007. 100

[215] E. Marder-Eppstein, E. Berger, T. Foote, B. P. Gerkey, and K. Konolige, "The office marathon: Robust navigation in an indoor office environment," in *International Conference on Robotics and Automation*, 05/2010 2010. 101

[216] M. Kalandros, L. Trailovic, L. Pao, and Y. Bar-Shalom, "Tutorial on multisensor management and fusion algorithms for target tracking," in *American Control Conference, 2004. Proceedings of the 2004*, vol. 5, 30 2004-july 2 2004, pp. 4734 –4748 vol.5. 102

[217] L. Andersson, "Multi-robot information fusion: Considering spatial uncertainty models," Ph.D. dissertation, Department of Management and Engineering, Linköping University, Sweden, 2008. 102

[218] D. Macii, A. Boni, M. De Cecco, and D. Petri, "Tutorial 14: multisensor data fusion," *Instrumentation Measurement Magazine, IEEE*, vol. 11, no. 3, pp. 24 –33, june 2008. 102

[219] A. Oswalt and M. Dombeck. Attention, memory and meta-cognition. (Last Accessed: 15th February 2013). [Online]. Available: http://www.sevencounties.org/poc/view_doc.php?type=doc&id=37681&cn=1272 104, 107

[220] D. Whitley, S. Rana, and R. B. Heckendorn, "The island model genetic algorithm: On separability, population size and convergence," *Journal of Computing and Information Technology*, vol. 7, pp. 33–47, 1998. 105

[221] Climbing robots. [Online]. Available: http://robotionary.com/search/hong+kong+the+chinese+university+tree+climbing+robot 117

[222] R. Siegwart and I. R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*. Scituate, MA, USA: Bradford Company, 2004. 118

[223] A. De Luca, G. Oriolo, and M. Vendittelli, *Control of Wheeled Mobile Robots: An Experimental Overview*. Springer-Verlag Heidelberg, 2001, vol. 270/2001, pp. 181–226. 118

[224] T.A.Baede, "Motion control of an omni directional mobile robot," Tech. Rep., 2006. 118

[225] L. R. G. Carrillo, A. E. D. López, R. Lozano, and C. Pégard, "Combining stereo vision and inertial navigation system for a quad-rotor uav," *Journal of Intelligent and Robotic Systems*, vol. 65, no. 1-4, pp. 373–387, 2012. 119

[226] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L. Cooper, M. Quigley, J. A. Adams, and C. Humphrey, "Supporting wilderness search and rescue using a camera-equipped mini uav: Research articles," *J. Field Robot.*, vol. 25, no. 1-2, pp. 89–110, Jan. 2008. [Online]. Available: http://dx.doi.org/10.1002/rob.v25:1/2 119, 120

[227] M. Andriluka, P. Schnitzspan, J. Meyer, S. Kohlbrecher, K. Petersen, O. von Stryk, S. Roth, and B. Schiele, "Vision based victim detection from unmanned aerial vehicles," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, oct. 2010, pp. 1740 –1747. 119, 120

[228] J. H. Choi, D. Lee, and H. Bang, "Tracking an unknown moving target from uav: Extracting and localizing an moving target with vision sensor based on optical flow," in *Automation, Robotics and Applications (ICARA), 2011 5th International Conference on*, dec. 2011, pp. 384 –389. 119

[229] J. Riehl, G. Collins, and J. Hespanha, "Cooperative search by uav teams: A model predictive approach using dynamic graphs," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 47, no. 4, pp. 2637 –2656, october 2011. 119

[230] D. Kingston, R. Beard, and R. Holt, "Decentralized perimeter surveillance using a team of uavs," *Robotics, IEEE Transactions on*, vol. 24, no. 6, pp. 1394 –1404, dec. 2008. 119

[231] K. V. Stefanik, J. C. Gassaway, K. Kochersberger, and A. L. Abbott, "Uav-based stereo vision for rapid aerial terrain mapping," *GIScience Remote Sensing*, vol. 48, no. 1, pp. 24–49, 2011. 119

[232] S. Weiss, M. W. Achtelik, M. Chli, and R. Siegwart, "Versatile distributed pose estimation and sensor self-calibration for an autonomous mav," in *Proceedings of*

the IEEE International Conference on Robotics and Automation (ICRA), 2012. 119

[233] Y.-c. Liu and Q.-h. Dai, "A survey of computer vision applied in aerial robotic vehicles," in *Optics Photonics and Energy Engineering (OPEE), 2010 International Conference on*, vol. 1, may 2010, pp. 277 -280. 119

[234] M. Achtelik, S. Weiss, M. Chli, F. Dellaert, and R. Siegwart, "Collaborative stereo," in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2011. 119

[235] M. Taylor, "At sea with underwater robots," 2010. [Online]. Available: http://spectrum.ieee.org/automaton/robotics/industrial-robots/at-sea-with-underwater-robots 120

[236] Hiteshk, "Irobot: Advent of the unmanned underwater vehicle (uuv)," 2010. [Online]. Available: http://www.marineinsight.com/sports-luxury/futuristic-shipping/irobot-advent-of-the-unmanned-underwater-vehicle-uuv/ 120

[237] K. Yoshida, "Achievements in space robotics," *Robotics Automation Magazine, IEEE*, vol. 16, no. 4, pp. 20 -28, december 2009. 120

[238] J. Borenstein, H. R. Everett, L. Feng, and D. Wehe, "Mobile robot positioning sensors and techniques," 1997. 121

[239] R. C. Gonzalez and R. E. Woods, *Digital image processing*, 3rd ed. Upper Saddle River, NJ: Prentice-Hall, 2008. 121

[240] D. Pascale, "A review of rgb color spaces...from xyY to R'G'B'," Tech. Rep., 2003. 121

[241] A. C. Murillo, J. J. Guerrero, and C. Sagus, "Topological and metric robot localization through computer vision techniques," in *Unifying Perspectives in Computational and Robot Vision*, ser. Lecture Notes in Electrical Engineering, D. Kragic and V. Kyrki, Eds. Springer US, 2008, vol. 8, pp. 113-128. 122, 129

[242] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 1, pp. 652-659, 2004. 122

[243] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis, "Monocular visual odometry in urban environments using an omnidirectional camera," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, 2008, pp. 2531 -2538. 122

[244] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat, "Visual topological slam and global localization," in *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, ser. ICRA'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 2029–2034. [Online]. Available: http://portal.acm.org/citation.cfm?id=1703775.1703780 122

[245] Egomotion. [Online]. Available: http://en.wikipedia.org/wiki/Egomotion 122

[246] X. Armangue, H. Araujo, and J. Salvi, "A review on egomotion by means of differential epipolar geometry applied to the movement of a mobile robot," vol. 36, no. 12, pp. 2927–2944, December 2003. 122

[247] T. Tuytelaars and K. Mikolajczyk, *Local Invariant Feature Detectors: A Survey*. Hanover, MA, USA: Now Publishers Inc., 2008. 122, 124

[248] I. Ulrich and I. Nourbakhsh, "Appearance-based place recognition for topological localization," in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 2, 2000, pp. 1023 –1029 vol.2. 123, 130

[249] C. Harris and M. Stephens, "A Combined Corner and Edge Detection," in *Proceedings of The Fourth Alvey Vision Conference*, 1988, pp. 147–151. 125

[250] J. Shi and C. Tomasi, "Good features to track," in *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, 1994, pp. 593 – 600. 125

[251] K. Mikolajczyk and C. Schmid, "Scale & affine invariant interest point detectors," *Int. J. Comput. Vision*, vol. 60, pp. 63–86, October 2004. 125

[252] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004. 125

[253] S. Se, D. G. Lowe, and J. Little, "Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks," in *International Journal of Robotics Research, 21, 8*, 2002, pp. 735–758. 125

[254] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346 – 359, 2008. 125

[255] J.-Y. Bouguet, "Pyramidal implementation of the lucas kanade feature tracker description of the algorithm," 2000. 125

[256] A. E. Abdel-Hakim and A. A. Farag, "Csift: A sift descriptor with color invariant characteristics," in *Proceedings of the 2006 IEEE Computer Society Conference*

on Computer Vision and Pattern Recognition - Volume 2, ser. CVPR '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 1978–1983. 125

[257] L. Juan and O. Gwon, "A Comparison of SIFT, PCA-SIFT and SURF," *International Journal of Image Processing (IJIP)*, vol. 3, no. 4, pp. 143–152, 2009. 125

[258] A. Bosch, A. Zisserman, and X. Munoz, "Scene classification using a hybrid generative/discriminative approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, 2008. 127

[259] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2, no. 6, pp. 559–572, 1901. 127

[260] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, ser. SIGIR '99. New York, NY, USA: ACM, 1999, pp. 50–57. 127

[261] T.-W. Lee, M. Lewicki, and T. Sejnowski, "Ica mixture models for unsupervised classification of non-gaussian classes and automatic context switching in blind signal separation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 10, pp. 1078 – 1089, oct 2000. 127

[262] A. Bakhtiari, "Analysis of the dirichlet process mixture model with application to dialogue act classification," Master's thesis, Département d'informatique et de génie logiciel, Faculté des sciences et de génie, Université Laval, Québec, 2011. 127

[263] P. Angelov and X. Zhou, "Evolving fuzzy-rule-based classifiers from data streams," *Fuzzy Systems, IEEE Transactions on*, vol. 16, no. 6, pp. 1462 – 1475, 2008. 128

[264] A. Blum, "On-line algorithms in machine learning," in *In Proceedings of the Workshop on On-Line Algorithms, Dagstuhl*. Springer, 1996, pp. 306–325. 128, 129

[265] J. Kivinen and M. K. Warmuth, "The perceptron algorithm vs. winnow: linear vs. logarithmic mistake bounds when few input variables are relevant," in *Proceedings of the eighth annual conference on Computational learning theory*, ser. COLT '95. New York, NY, USA: ACM, 1995, pp. 289–296. 128, 129

[266] N. Littlestone, "Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm," *Mach. Learn.*, vol. 2, no. 4, pp. 285–318, Apr. 1988. 129

[267] E. Menegatti, T. Maeda, and H. Ishiguro, "Image-based memory for robot navigation using properties of omnidirectional images," *Robotics and Autonomous Systems*, vol. 47, no. 4, pp. 251 – 267, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/B6V16-4CK7XKN-1/2/7741d06cc795d217503bc7f77a186c6e 129, 130

[268] J. Courbon, G. Blanc, Y. Mezouar, and P. Martinet, "Navigation of a non-holonomic mobile robot with a memory of omnidirectional images," in *Workshop on Planning, Perception and Navigation for Intelligent Vehicles, ICRA'07*, Roma, Italy, April 2007, pp. 51–56. 129

[269] M. Cummins and P. Newman, "Probabilistic appearance based navigation and loop closing," in *Proc. IEEE International Conference on Robotics and Automation (ICRA'07)*, Rome, April 2007. 130, 142

[270] Z. Wang, L. Liu, and M. Zhou, "Protocols and applications of ad-hoc robot wireless communication networks: An overview," *International Journal of Intelligent Control and Systems*, vol. Vol.10, No.4, pp. 296–303, Dec 2005. 131, 132, 133

[271] W. Sheng, Q. Wang, Q. Yang, and S. Zhu, "Minimizing data exchange in ad hoc multi-robot networks," in *Advanced Robotics, 2005. ICAR '05. Proceedings., 12th International Conference on*, july 2005, pp. 811 –816. 132

[272] R. Luo and T. M. Chen, "Development of a multi-behavior based mobile robot for remote supervisory control through the internet," *Mechatronics, IEEE/ASME Transactions on*, vol. 5, no. 4, pp. 376 –385, dec 2000. 133

[273] K. LeBlanc and A. Saffiotti, "Cooperative information fusion in a network robot system," in *Proceedings of the 1st international conference on Robot communication and coordination*, ser. RoboComm '07. Piscataway, NJ, USA: IEEE Press, 2007, pp. 42:1–42:4. [Online]. Available: http://dl.acm.org/citation.cfm?id=1377868.1377920 133

[274] R. Lundh, L. Karlsson, and A. Saffiotti, "Autonomous functional configuration of a network robot system," *Robotics and Autonomous Systems*, vol. 56, no. 10, pp. 819 – 830, 2008, ¡ce:title¿Network Robot Systems¡/ce:title¿. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889008000857 133

[275] U. Witkowski, M. El-Habbal, S. Herbrechtsmeier, A. Tanoto, J. Penders, L. Alboul, and V. Gazi, "Ad-hoc network communication infrastructure for multirobot systems in disaster scenarios," in *IARP/EURON Workshop on Robotics for Risky Interventions and Environmental Surveillance*, 2008. 133

[276] H. Sugiyama, T. Tsujioka, and M. Murata, "Autonomous chain network formation by multi-robot rescue system with ad hoc networking," in *Safety Security and Rescue Robotics (SSRR), 2010 IEEE International Workshop on*, july 2010, pp. 1 -6. 133

[277] T. Ramrekha, E. Panaousis, and C. Politis, "Standardisation advancements in the area of routing for mobile ad-hoc networks," *The Journal of Supercomputing*, pp. 1-26, 2011, 10.1007/s11227-011-0705-2. [Online]. Available: http://dx.doi.org/10.1007/s11227-011-0705-2 133

[278] L. Sentis, M. Mintz, A. Ayyagari, C. Battles, S. Ying, and O. Khatib, "Large scale multi-robot coordination under network and geographical constraints," in *Industrial Electronics, 2009. ISIE 2009. IEEE International Symposium on*, july 2009, pp. 1046 -1053. 135

[279] G. Biggs and B. Macdonald, "A survey of robot programming systems," in *in Proceedings of the Australasian Conference on Robotics and Automation, CSIRO*, 2003, p. 27. 135, 136

[280] T. Schneider, "Distributed networks using ros cross-network middleware communication using ipv6," Master's thesis, Department of Electrical Engineering and Information Technology, Technische Universität München (TUM), 2012. 136, 137

[281] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009. 136

[282] D. Meger, M. Muja, S. Helmer, A. Gupta, C. Gamroth, T. Hoffman, M. Baumann, T. Southey, P. Fazli, W. Wohlkinger, P. Viswanathan, J. Little, D. Lowe, and J. Orwell, "Curious george: An integrated visual search platform," in *Computer and Robot Vision (CRV), 2010 Canadian Conference on*, 31 2010-june 2 2010, pp. 107 -114. 136, 137

[283] S. Cousins, "Exponential growth of ros [ros topics]," *Robotics Automation Magazine, IEEE*, vol. 18, no. 1, pp. 19 -20, march 2011. 137

[284] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 99th ed. Wiley-Interscience, August 1991. 144

[285] Videre Design LLC, *ERA Mobile Robot Users Manual*, Rev H ed., October 2009. 150

Index