Topological scene recognition and visual navigation for mobile robots using omnidirectional camera

Author(s):
Liu, Ming

Publication Date:
2013

Permanent Link:
https://doi.org/10.3929/ethz-a-010005794

Rights / License:
In Copyright - Non-Commercial Use Permitted
DISS. ETH NO. 21279

Topological Scene Recognition and Visual Navigation for Mobile Robots using Omnidirectional Camera

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH

(Dr. sc. ETH Zurich)

presented by

MING LIU

Bachelor of Engineering
Tongji University
Shanghai, China

Date of birth
30.03.1984

citizen of
China

accepted on the recommendation of

Prof. Dr. R. Siegwart, Supervisor
Prof. Dr. M. Vincze, Co-examiner

2013
Acknowledgement

Firstly, I would like to express my thanks to Prof. Roland Siegwart, the head of the Autonomous System Laboratory (ASL) of ETH Zürich, without whom I would never have the chance to become a member of this one of the top robotics laboratories in the world. He deserves my deepest gratitude for taking me into the group, for providing excellent facilities, and the wonderful inspiring atmosphere. Furthermore, to give me these highly appreciated experience which led to many happy memories.

Furthermore, I would like to thank all the great people I have met in the lab, specially to Francis Colas, Cedric Pradalier and Davide Scaramuzza. I owe a lot to you, not only because of the chance to work with you great scientists, rarely intelligent and, no secondary, good friends, also for helping me to start the very first steps into the research field. Your deep insights of probabilistic techniques, system integration, computer vision and other extended fields deeply attract me to always have the courage to move one step further.

A special mention goes to my Canadian friend Francois Pomerleau, who is specially supportive during all the integration tasks and evaluations, and a sharp minded guy with whom you can talk about anything. I would also thank a lot of other people from the lab, which I may not be able to show the long list of their names, for the interesting discussions in general robotics, helps in life and much more, especially for the fun we had together, Gregory Hitz, Stephane Magnenat, Stephan Weiss, Laurent Kneip, Marco Hutter, Jiwon Shin and many more.

I am honored to have Prof. Markus Vincze, the coordinator of robots@home EU project. Your support as being co-evaluator is highly appreciated.

I would like to thank my father, my mother and my family for the support I always received from their positive attitude.

This work would be impossible without the support of two main European projects that I had the pleasure and opportunity to take part in:

- **Natural Human-Robot Cooperation in Dynamic Environments (NIFTi)**
  http://www.nifti.eu
NIFTi is about teams of robots and humans doing tasks together, trying to reach a shared goal. NIFTi looks at how the robot could bear human agents in mind. NIFTi puts the human factor into cognitive robots.
Each year, NIFTi evaluates its systems together with several USAR organizations. Rescue personnel teams up with NIFTi robots to carry out realistic missions, in real-life training areas.

- **Robots@Home**  
http://robots-at-home.acin.tuwien.ac.at

The objective of robots@home is to provide an open mobile platform for the massive introduction of robots into the homes of everyone. The innovations will be: (1) A scaleable, affordable platform in response to the different application scenarios of the four industrial partners: domotics, security, food delivery, and elderly care. (2) An embedded perception system providing multi-modal sensor data for learning and mapping of the rooms and classifying the main items of furniture. (3) A safe and robust navigation method that finally sets the case for using the platform in homes everywhere. The system was tested in four homes and at a large furniture store.
Abstract

Mobile robots are intelligent agents that are designed to serve people in various circumstances. With time, more and more robots will share the surrounding environments with humans, for example, service robots in families and autonomous cars on the road etc. In all these cases above, it is important that the robots are able to provide an efficient representation of the environment, which on one hand can be used as a common understanding among all involved subjects in the scenario; on the other hand, can support the robots to fulfill assigned missions.

Psychological works demonstrated that humans can understand their surroundings by mostly topological meanings. I adopt this concept all across this thesis. An example can be observed as for service robots. In most situations, human may like to control the house-hold robot by naming - go to “kitchen” or “grandpa’s bed”. Topological modeling of the environment would greatly help these applications. Moreover, navigation among topological nodes in the environment is essential to carry out practical services. Based on these observations, with this thesis, I present several contributions in both the fields of robotic scene recognition and visual navigation.

In the first part, I consider the environment modeling using an omnidirectional camera as the only sensor. I present a topological scene recognition algorithm, which is modeled by a Dirichlet Process Mixture Model (DPMM). Topological segments with a similar appearance in video sequences can be automatically clustered and recognized in real-time. In the second part, regarding the navigation among topological nodes, waypoint-based topological modeling of the environment is presented, centered on a visual homing framework using Image-Based Visual Servoing (IBVS). A finite state-machine is then adopted, which enables topological navigation between waypoints by fusing Visual Homing and odometry motion. Experiments on datasets and real environments show the competence of the proposed algorithms against state-of-the-art.

Key words: Scene Recognition, Visual Homing, Topological Visual Navigation, Non-parametric Learning, Mobile Robotics
Kurzbeschreibung

Mobile Roboter sind intelligente Agenten, die mit dem Ziel entworfen werden, Menschen in verschiedenen Situationen zu dienen. Mit der Zeit werden mehr und mehr Roboter die Umwelt mit Menschen teilen; so zum Beispiel Serviceroboter im Haushalt und autonome Fahrzeuge auf im Verkehr. In allen Fällen ist es wichtig, dass die Roboter in der Lage sind, eine effiziente abstrakte Beschreibung der Umgebung zu generieren. Einerseits, damit diese als gemeinsames Grundlage für alle beteiligten Systeme im Szenario verwendet werden kann; andererseits, damit die Roboter sie unterstützend beim ausüben einer spezifischen Aufgabe verwenden können.

# Contents

Acknowledgement ................................................................. i  
Abstract ............................................................................... iii  
Kurzbeschreibung ................................................................. v  
Acronyms ............................................................................... xi  
Contents ................................................................................ x  
List of Figures ......................................................................... xvi  
List of Tables ............................................................................ xvii  
0 Introduction ......................................................................... 1  
0.1 Robotics and Perception ................................................... 1  
  0.1.1 Sensors and sensing ...................................................... 3  
  0.1.2 Robotic Platforms ....................................................... 5  
  0.1.3 Perception and Representations .................................... 6  
0.2 How do humans recognize the “world”? ............................ 7  
0.3 Robotic Environment Modeling ......................................... 9  
0.4 Dissertation Outline and Contributions ............................. 10  
  0.4.1 Visual topological scene recognition ............................. 10  
  0.4.2 Visual topological navigation ....................................... 11  
Part I: Visual Topological Scene Recognition ........................... 15  
1 Information Fusion for Simultaneously Clustering and Recognition 17  
  1.1 Introduction .................................................................... 17  
    1.1.1 From perception to recognition .................................... 17  
    1.1.2 From recognition to clustering .................................... 17  
    1.1.3 Simultaneous clustering and recognition .................... 19  
  1.2 Information Fusion and Perception ................................... 19  
    1.2.1 Multi Information Fusion .......................................... 19  
    1.2.2 Clustering ............................................................... 21  
    1.2.3 Recognition and inference ........................................ 21  
    1.2.4 Assumptions and Contributions ............................... 22  
  1.3 Model Formulation ......................................................... 23  
    1.3.1 Model of Information Fusion ..................................... 23  
    1.3.2 Multi Information Perception ................................. 26  
    1.3.3 Model Inference ...................................................... 26  
  1.4 Approximation ............................................................... 28  
  1.5 Simulation ...................................................................... 29  
  1.6 Discussion and Summary ................................................ 31
2 Fast Adaptive Color Tag (FACT) .............................................. 33
  2.1 Scene Recognition with an Omnidirectional Camera ............ 33
    2.1.1 Scene descriptors ............................................ 33
  2.2 Formation of FACT .................................................. 35
    2.2.1 Unwrapping of Omnidirectional Image ...................... 35
    2.2.2 Segmentation of the Panorama ............................... 35
    2.2.3 FACT Descriptor and Topological mapping .................. 36
    2.2.4 Construction of FACT Descriptor ............................ 37
  2.3 Naive Matching algorithm ........................................... 38
    2.3.1 Test of robustness in dynamic environment ............... 42
    2.3.2 Drawbacks of Naive Matching algorithm .................... 43
3 DP-FACT: Scene recognition with DPMM ............................... 45
  3.1 Modeling in term of Simultaneous Clustering and Recognition
    (SCAR) .................................................................. 45
  3.2 Matching of DP-FACT .................................................. 47
    3.2.1 DF-FACT Formulation ......................................... 47
    3.2.2 Non-parametric test .......................................... 48
    3.2.3 Model update .................................................. 49
    3.2.4 Pair-wise Distance for DP-FACT .............................. 50
4 Experiments and Discussion ............................................... 55
  4.1 Comparison in Accuracy ............................................... 55
  4.2 Evaluation in time cost ............................................... 58
  4.3 Discussion on χ² test and naive matching ....................... 59
  4.4 Further experiments on public dataset ........................... 61
5 Summary ...................................................................... 65

Part II: Visual Topological Navigation .................................. 67
6 Visual homing and visual servoing .................................... 69
  6.1 Visual Servoing Framework ......................................... 69
    6.1.1 Principles of Visual Servoing ............................... 69
    6.1.2 Visual Homing .................................................. 70
    6.1.3 Integration and Control Loop ............................... 72
    6.1.4 Visual homing by omnidirectional cameras ............... 73
  6.2 Image Based Visual Servoing ........................................ 73
    6.2.1 Definitions of Visual Error ................................. 74
    6.2.2 Error derivative .............................................. 75
  6.3 Observability Analysis ................................................ 77
    6.3.1 Definitions .................................................... 77
    6.3.2 Observation matrix .......................................... 80
7 Fast Visual Homing: SSVS ................................................. 83
  7.1 Scale-based Control for a 1-D Robot ............................... 83
7.2 Generic Scale-based Control .................................................. 84
7.3 Outlier rejection ................................................................. 85

8 Simulation results for visual homing ........................................ 87
  8.1 Performance comparison using image-based visual homing ...... 87
    8.1.1 Comparison of all methods on a particular example .......... 87
    8.1.2 Statistical comparison of the different approaches .......... 90
    8.1.3 Success rate .......................................................... 91
    8.1.4 Final position error .................................................. 92
    8.1.5 Number of iterations ............................................... 94
    8.1.6 Influence of feature distribution ................................. 94
  8.2 Influence of the assumptions for SSVS ............................... 95
    8.2.1 Influence of known distance assumption ...................... 95
    8.2.2 Influence of noise ................................................. 96
  8.3 Parametrization of the outlier rejection ............................. 98

9 Experiments and Discussion ................................................ 101
  9.1 Node validation ............................................................ 101
  9.2 Results on an indoor data-set ......................................... 103
  9.3 Results of real-time homing test .................................... 108
    9.3.1 Homing vectors to one reference ............................... 108
    9.3.2 Experiment of outlier rejection ................................. 109

10 Navigation Design ........................................................... 113
  10.1 Overview ................................................................. 113
  10.2 Mapping ................................................................. 114
  10.3 Localization in the map ............................................... 115
  10.4 Navigation .............................................................. 116
  10.5 Visual Compass ......................................................... 117
  10.6 Experiments and results ............................................... 119
  10.7 Functional test .......................................................... 121
  10.8 Longer time roaming test .............................................. 123

11 Summary ................................................................. 125

Conclusion and Future Work .................................................. 127
12 Conclusion ................................................................. 129
  12.1 Summary ................................................................. 129
  12.2 Outlook ................................................................. 133
  12.3 Closing comments ...................................................... 134

Bibliography ................................................................. 149
Curriculum Vitæ .............................................................. 151


**Acronyms**

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPMM</td>
<td>Dirichlet Process Mixture Model</td>
</tr>
<tr>
<td>IBVS</td>
<td>Image-Based Visual Servoing</td>
</tr>
<tr>
<td>BOVS</td>
<td>Bearing-Only Visual Servoing</td>
</tr>
<tr>
<td>SOVS</td>
<td>Scale-Only Visual Servoing</td>
</tr>
<tr>
<td>SBVS</td>
<td>Compound Scale-Bearing Visual Servoing</td>
</tr>
<tr>
<td>HSVS</td>
<td>Heuristic Scale-based Visual Servoing</td>
</tr>
<tr>
<td>SSVS</td>
<td>Simplified Scale-based Visual Servoing</td>
</tr>
<tr>
<td>USAR</td>
<td>Urban Search And Rescue</td>
</tr>
<tr>
<td>FOV</td>
<td>Field Of View</td>
</tr>
<tr>
<td>SfM</td>
<td>Structure from Motion</td>
</tr>
<tr>
<td>LIDAR</td>
<td>Ligh Detection And Ranging</td>
</tr>
<tr>
<td>RADAR</td>
<td>Radio Detection And Ranging</td>
</tr>
<tr>
<td>SLAM</td>
<td>Simultaneous Localization And Mapping</td>
</tr>
<tr>
<td>FACT</td>
<td>Fast Adaptive Color Tags</td>
</tr>
<tr>
<td>SCAR</td>
<td>Simultaneous Clustering and Recognition</td>
</tr>
</tbody>
</table>
When a system is completely defined, some damn fool discovers something which either abolishes the system or expands it beyond recognition.

Unknown
# List of Figures

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1.1</td>
<td>Several examples of service mobile robots</td>
<td>2</td>
</tr>
<tr>
<td>0.1.2</td>
<td>Several examples of robots for rescue purpose</td>
<td>3</td>
</tr>
<tr>
<td>0.1.3</td>
<td>Imaging sensors and range finders</td>
<td>4</td>
</tr>
<tr>
<td>0.1.4</td>
<td>Examples showing pros and cons of imaging sensor and range finder</td>
<td>5</td>
</tr>
<tr>
<td>0.1.5</td>
<td>Collection of Differential-driven Robots</td>
<td>6</td>
</tr>
<tr>
<td>0.2.1</td>
<td>Patterns humans use to model the world</td>
<td>8</td>
</tr>
<tr>
<td>1.1.1</td>
<td>Recognition of the can of cola by information fusion</td>
<td>18</td>
</tr>
<tr>
<td>1.1.2</td>
<td>Recognition paradigm by information fusion</td>
<td>18</td>
</tr>
<tr>
<td>1.1.3</td>
<td>Simultaneously clustering and recognition of drinks</td>
<td>19</td>
</tr>
<tr>
<td>1.3.1</td>
<td>Directed Acyclic Graph (DAG) of the generation of a single observation</td>
<td>23</td>
</tr>
<tr>
<td>1.3.2</td>
<td>Chinese Restaurant Process</td>
<td>24</td>
</tr>
<tr>
<td>1.3.3</td>
<td>Directed Acyclic Graph (DAG) of the proposed model for recognition by multi information fusion</td>
<td>25</td>
</tr>
<tr>
<td>1.5.1</td>
<td>Simulated multi sensor readings</td>
<td>30</td>
</tr>
<tr>
<td>1.5.2</td>
<td>Likelihood of readings</td>
<td>31</td>
</tr>
<tr>
<td>1.5.3</td>
<td>Simulation of change point detection and clustering result</td>
<td>32</td>
</tr>
<tr>
<td>2.1.1</td>
<td>Omnidirectional camera and panoramic image</td>
<td>34</td>
</tr>
<tr>
<td>2.2.1</td>
<td>All the circles detected in the raw image. The outermost circle is extracted.</td>
<td>35</td>
</tr>
<tr>
<td>2.2.2</td>
<td>A unwrapped result</td>
<td>36</td>
</tr>
<tr>
<td>2.2.3</td>
<td>An output of vertical edges detection</td>
<td>36</td>
</tr>
<tr>
<td>2.2.4</td>
<td>The segmentation result</td>
<td>36</td>
</tr>
<tr>
<td>2.2.5</td>
<td>The segmentation process</td>
<td>37</td>
</tr>
<tr>
<td>2.3.1</td>
<td>A schematic diagram of Test 1 during the matching process</td>
<td>39</td>
</tr>
<tr>
<td>2.3.2</td>
<td>Distance matrix for the pair-wise beliefs</td>
<td>41</td>
</tr>
<tr>
<td>2.3.3</td>
<td>An example taken in the corridor, showing the effect of moving objects/human.</td>
<td>43</td>
</tr>
</tbody>
</table>
2.3.4 The test result of the robustness of FACT. The red line indicates the threshold \( TH_{\text{global}} \) which defines the matching status of current image.

3.1.1 System Model

3.2.1 Example of Histograms over discretized UV space

3.2.2 Pair-wise distance matrix in UV-color and width space

3.2.3 Distance matrix using combined features for different \( \rho \) values

3.2.4 Resulting distance matrix using a median filtering

5 Sample images for the omnidirectional camera dataset

4.1.1 Experiment results. From top to bottom: raw label output of DP-FACT; result of DP-FACT after median voting filter of 5 frames; result of keypoint-based approaches (SIFT, BRISK) after median filtering; image sequence in a compressed layout; labeled ground truth; result of DP-FACT; label explanations; an overlaid sketch of the test environment by detected scene appearances.

4.2.1 Time cost of \( \text{DP-FACT} \) over frames. The lines are filtered results out of raw measurements (in circles). The gray area indicates the inference time.

4.2.2 Inference time vs number of nodes

4.2.3 Time cost comparison

4.2.4 Performance on different CPUs, using single core

4.4.1 Segmentation result based on \( \text{DP-FACT} \) for Freiburg Path A [1]

6.1.1 The control loop of visual homing

6.1.2 Abstracted problem of homing. Keypoints \( p_1 \) to \( p_n \) can be observed at the current position \( C \) and at the Home position \( O \). The variant sizes indicate the differences of key-points in scale.

7.3.1 A result of the outlier rejection algorithm. The gray linkages mark the inliers of the matching; white for outliers.

8.1.1 Left: 45 degree view and top view of the simulated environment and trajectories of four different methods. Magenta: Bearing-Only Visual Servoing (BOVS), Red: Scale-Only Visual Servoing (SOVS), Cyan: Compound Scale-Bearing Visual Servoing (SBVS), Black: Simplified Scale-based Visual Servoing (SSVS). Right: Error evaluation on the simulated environment referring to (20,0)
### Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1.2</td>
<td>The visual feature errors and error distance to the home position over time</td>
<td>89</td>
</tr>
<tr>
<td>8.1.3</td>
<td>Statistical simulation setup</td>
<td>91</td>
</tr>
<tr>
<td>8.1.4</td>
<td>Final position errors</td>
<td>92</td>
</tr>
<tr>
<td>8.1.5</td>
<td>Number of iteration required against the initial distance to the home position</td>
<td>93</td>
</tr>
<tr>
<td>8.1.6</td>
<td>Trajectory resulting from a bad feature distribution, and corresponding error analysis</td>
<td>95</td>
</tr>
<tr>
<td>8.2.1</td>
<td>Comparison between the distances to features are known and unknown. Green: known; Red: Unknown</td>
<td>96</td>
</tr>
<tr>
<td>8.2.2</td>
<td>Trajectory on noisy observation, and error analysis. The error analysis contains the trends of extra length traveled, final position error and iterations required over different noise levels. All the units are related to the simulation environment.</td>
<td>97</td>
</tr>
<tr>
<td>8.3.1</td>
<td>Evaluation of the assumption for outlier rejection</td>
<td>99</td>
</tr>
<tr>
<td>9.1.1</td>
<td>Histograms of the feature distribution. Green features have been kept after the outlier rejection stage. Red features have been rejected.</td>
<td>102</td>
</tr>
<tr>
<td>9.2.1</td>
<td>Homing vectors and error analysis referring to (5.8). The color-map in the first row indicates the number of matched key-points; in the second row, the color-map indicates the average angular error in degree.</td>
<td>104</td>
</tr>
<tr>
<td>9.2.2</td>
<td>Average Error and processing time with different resolution levels</td>
<td>106</td>
</tr>
<tr>
<td>9.2.3</td>
<td>Relation between AAE and the distribution of features</td>
<td>108</td>
</tr>
<tr>
<td>9.3.1</td>
<td>Sample image of the data-set with arbitrarily moving objects</td>
<td>108</td>
</tr>
<tr>
<td>9.3.2</td>
<td>Results of SSVS with moving people in the scene</td>
<td>109</td>
</tr>
<tr>
<td>9.3.3</td>
<td>Histogram of angular errors for the selected algorithms under the test condition with moving people</td>
<td>110</td>
</tr>
<tr>
<td>9.3.4</td>
<td>Error comparison in the cases of with/without outlier rejection. The labels marked with _on indicate the boxplot is with outlier rejection, _off indicates the results calculated from the raw matching result.</td>
<td>111</td>
</tr>
<tr>
<td>10.1.1</td>
<td>An application instance in a 130m² apartment at Vienna Opern-</td>
<td>113</td>
</tr>
<tr>
<td>10.1.2</td>
<td>The system structure</td>
<td>114</td>
</tr>
<tr>
<td>10.1.3</td>
<td>A sketch of the state machine of the navigation system.</td>
<td>115</td>
</tr>
<tr>
<td>10.4.1</td>
<td>The flow chart of the general positioning process</td>
<td>117</td>
</tr>
<tr>
<td>10.4.2</td>
<td>The flow chart of the pose stabilization process</td>
<td>118</td>
</tr>
</tbody>
</table>
List of Tables

4.1.1 Accuracy of scene recognition ........................................ 58
4.4.1 Test result using COLD datasets ..................................... 63
9.2.1 Error Analysis for all the algorithms (in degree) ............... 103
9.2.2 Effect of image resolution on the AAE referring to (5,8). .... 107
Introduction

The whole of science is nothing more than a refinement of everyday thinking.

Albert Einstein

This dissertation addresses fundamental problems concerning visual topological modeling of environments for mobile robots using an omnidirectional camera. It aims at making robots serve humans in a flexible way. This chapter gives an introduction to the context by starting from general applications and scientific perspectives of mobile robots, then focusing on the nowadays problems, emphasizing the aspects conducted in this thesis.

0.1 Robotics and Perception

Robotic research and application have drawn great attention in the past decades. Since the birth of the first intelligent machine, robots have served human society from various aspects. Nowadays, we could see that an essential transition is happening. Robots are not just an extension of humans’ work force in industrial assembly lines. With the rapid development of artificial intelligence, mechatronics, material science and sensorial technologies, they have been a part of our daily life. From various aspects, robots have even changed our way of living. Several reported applications have shown their capabilities in house work [2], tour guide [3], shopping [4], entertainment [5], waiter service [6], education [7, 8] and so on. A collection of representatives can be found in Figure 0.1.1. All the robots listed here can be categorized as service robots in general. It is preferred for such robots to easily understand human intentions and be easily understood. Since humans have mostly a topological understanding of the surrounding environment [9], it will be a great help when such a similar representation can be granted to robots.
2. INTRODUCTION

(a) Roomba Cleaning Robot (b) FP7 EUROPA (c) Robox Robot (d) ThymioII

Figure 0.1.1: Several examples of service mobile robots

Before we go into details of scientific problems, let us first recall some basic rules for robotics itself. The famous science fiction author Isaac Asimov stated in 1942 that, later named as “Three Laws of Robotics”:

1. A robot may not injure a human being or, through inaction, allow a human being to come to harm.
2. A robot must obey the orders given to it by human beings, except where such orders would be conflict with the First Law.
3. A robot must protect its own existence as long as such protection does not conflict with the First or Second Laws.

It is widely referred that they are the most important regulations for all robotic applications. It can be inferred that all these laws firstly depend on how a robot understands the surrounding environment and situation. This matters a lot for the subsequent behaviors of the robot. We can call such capability of a robot as Environment understanding, which implies the process of Environment modeling, leading to the construction of an Environment model.

Technically, we could see that the first law requires more sophisticated intelligence than the others, since the core of the other two laws can be compensated by hardware or material development, and tight integration of control techniques. In the matter of fact, more and more real robotic systems have been developed to emphasize the first law. They have indeed shown the surprisingly extended capability beyond human beings in several emergency or extreme situations. Typically, the importance of their roles as irreplaceable team-members in Urban Search And Rescue (USAR) missions [10–13], especially for surveillance and device maintaining purposes, have been greatly exhibited. Several examples are shown in Figure 0.1.2. In USAR missions,
the common understanding of the environment is even more important, considering the requirement of prompt response from the rescue team.

For all these mentioned applications, the fundamental problem for robots is the utilization of various sensors mounted on the robot to sense, interpret and comprehend the surrounding environments. The union of these problems is generally summarized as the terminology perception.

### 0.1.1 Sensors and sensing

Perception always starts with sensing. In this thesis, omnidirectional camera is taken as the primary exteroceptive sensor. Regarding sophisticated sensors (excluding sensors with scalar output, e.g. humidity or temperature sensors), there are three main types are used by robotics: imaging sensors, range finders and intermediate sensors. In my opinion, these three sensors are intimately related as depicted in Figure [0.1.3](#).

**Imaging sensors**  Imaging sensors are commonly named as cameras. Intuitively, images are the primary outputs from such sensors, which can be video images, thermal images etc. Perspective monocular camera is the basic type of imaging sensors. Usually it has limited shutter speed, sensitive to illumination changes and white balance changes, and with limited Field Of View (FOV). Most of these drawbacks are embedded in the manufacturing process, except the FOV can be expanded by several means. Such instances that require particular calibration [14](#) are also presented in Figure [0.1.3](#). Ladybug™ combines 6 cameras looking at isotropic directions, outputting a stitched full 360° FOV. Another solution is to use a specially shaped mirror to reflect the environment to the perspective FOV. Regarding the complexity

---

[1](http://www.ptgrey.com/products/ladybug5)
of calibration, algorithms which are independent of full calibration of these cameras are generally preferred than others [15, 16]. The transition from imaging sensors to range estimation can possibly be obtained by triangulation [17] or Structure from Motion (SfM) [18]. Similarly, algorithms that do not require the full structural reconstruction are ideal in most cases [19, 20] due to lower computational complexity and data density.

**Range finder** In general, range finders can be mainly categorised as LIght Detection And Ranging (LIDAR) and RAdio Detection And Ranging (RADAR) based on their sensing fundamentals. The output of a range finder is a discretized distance map within the FOV of the sensor. With translation and rotation of a range finder, a three dimensional range image (or depth image) can be generated.

Two groups of examples are illustrated as Figure 0.1.4, which compares pros and cons of imaging sensors and range finders. I consider that imaging sensors and range finders have their own advantages in different situations. An extreme case is that a monocular camera cannot be used in a dark environment, which is common for USAR missions, whereas a LIDAR or RADAR can still be functioning. Further comparisons can be observed in Figure 0.1.4. We could see that for the case of (a)(b), it is not easy for an untrained user to tell (b) is a section of stairs, but the image representation in (a) is much...

---

**Figure 0.1.3:** Imaging sensors and range finders
informative. Sometimes visual information may provide false structural hints \[21\], e.g. (c) showing a standing cartoon character, however it is clearly not an object existing in 3D depicted by (d). It shows that additional analysis and refinement of the raw information are necessary, as the primary goal of perception.

**Intermediate Sensors** These sensors are usually able to provide both appearance and distance information. They are constructed by either a combination of various sensors e.g. Kinect\(^\text{TM}\) or introducing further geometrical constraints e.g. stereo vision system, in order to aid perception. For the first instance, multi-sensor fusion is generally used since the calibration between sensors is provided. On the other hand, stereo vision systems are mostly used for outdoor applications, as they are less sensitive to inferred noise and working distance can be tuned by adjusting the base-line size.

### 0.1.2 Robotic Platforms

In robotics, the sensors are usually mounted on mobile robot platforms. The experiments in this thesis are mainly carried out on differential-driven wheeled robots, which have relatively easy kinematics. Velocity commands can be directly applied with minor conversion to wheel speeds. Another advantage is that the odometry is more accurate than other types such as track robots, which generally simplifies algorithms by providing more reliable initial pose estimation. The differential-driven robots that I use for

\[\text{http://www.xbox.com/KINECT}\]
6 0. INTRODUCTION

(a) BIBA Robot  (b) James Robot  (c) Pioneer™ Robot (Differential four-wheel driven)

Figure 0.1.5: Collection of Differential-driven Robots

this dissertation are shown in Figure 0.1.5. Robots shown in (a)(b) can be flexibly adapted to various sensor types and positioning by modifying the profit-bar structures. Pioneer™ robot is a commercial product. It has two sets of wheels that are adaptive for indoor and outdoor environments respectively. Usually customized modification is required in order to use it for real applications. In Figure 0.1.5(c), the four-wheel model is depicted with extended sensor setup.

0.1.3 Perception and Representations

Driving the robots to have a reasonable understanding of the environment is not an easy task. From years of experience, we could see that perception is not only about the re-organizing or stacking of sensor readings. More important, for mobile robotics, it leads to a representation, a so-called Environment Model, within which the subject or with its cooperators can perform conducted tasks. In other words, an Environment Model is a set of representations that defines and indicates how a robot should understand its surroundings. Such a representation can be either dense or sparse.

http://www.mobilerobots.com/researchrobots/pioneerp3dx.aspx
Dense Representation defines characteristics of each point in the continuous work space of the robot. For example, an occupancy grid map \[22\] is a dense representation, since each point in the work space of the robot lies on a defined map and each point in the map is featured by its probability of occupancy.

Sparse Representation consists of extracted sparse features from the work space, i.e. using salient information to represent the environment. Feature-based navigation map, e.g. the maps defined by PTAM \[23\] or ARTags \[24\], and point-clouds are typical sparse representations.

Usually dense representation leads to larger size of data and memory cost, since it is a comprehensive representation of the work space. It also means that in most cases the resulted representation is redundant. Conversely, sparse representation is usually computationally more efficient with less data, however it is hard to reconstruct every detail of the environment. In order to complement this drawback, graph structures are commonly used, which define the nearest neighbour relations locally. As a result, information can be propagated or interpolated among neighbourhoods. In mobile robotics, both dense and sparse representations serve modeling of the work space. Generally, this modeling process is also named as Mapping.

In this thesis, the typical sparse representation - topological map, is considered. The key problems are inspired by observing how humans recognize the surrounding world, then extend the conducted concepts and representations to robotic applications, as shown in the next section.

### 0.2 How do humans recognize the “world”? 

The development of robotics is always inspired by learning from human activities. Before getting into the details of robotic mapping, we can first have a look at the several common ways for humans to model the world. In Figure 0.2.1, the common patterns are illustrated, taking the world map as an example. There are mainly three types of representations as follows.

1. **Metric Map**: this is the basic representation with objective ground-truth. It uses precise measurements in geometrical distance to describe the existence of entities. Generally speaking, geometrical precision is the only criteria for the quality of a metric map. Researchers of geography study for years to get the precise metric information as accurate as possible, generating maps such as Figure 0.2.1(a). The absolute coordinates in a metric map are usually the basis of other rep-
Figure 0.2.1: Patterns humans use to model the world
resentations. However, the metric measures, such as GPS coordinates, are usually less informative for human to understand the world.

2. **Appearance**: beside the exact measurements, another important hint we can use to model the world is the appearance. From the typical scenes in Figure 0.2.1(b) we could easily tell the differences and even recognize the places holding such appearances. For robotic applications, image processing techniques can be adopted to realize similar functionalities, which are usually conducted by scene recognition.

3. **Topological Waypoints**: typical applications using topological waypoints are shown as Figure 0.2.1(c). We can consider each city in the world as a topological waypoint, on the left of (c). A travel plan for flight is depicted among several major waypoints, indicating a “traveling scenario”. Another intuitive example is shown on the right side of (c), which depicts a public transportation map. Comparing with travelling among exact coordinates, we could see that topological waypoint-based navigation is a concise way to reach the goal and greatly simplifies our daily life. Similar concepts can be used by mobile robots for navigation purpose.

Inspired by these observations, this thesis is structured as two parts. The first part will deal with the appearance-based representation of the environment. Part II will focus on the visual navigation problem among waypoints.

### 0.3 Robotic Environment Modeling

Robotic Mapping is a widely studied problem, since it is the basis for almost all mobile robotics tasks. As far as mobile robot is concerned, it is coupled with *Localization*. After all, knowing “where am I?” seems to be a prior of “what does THIS place look like?”, however the answer to the first question needs a base representation derived from the latter as well.

As a delegate of mapping in robotics, Simultaneous Localization And Mapping (SLAM) aims at solving the two questions stated above simultaneously. Using sensor readings, it generates an environment representation and estimation of robot pose(s). If the resulting robot poses contain not only the current pose but also all history tracks, it is named full-SLAM. Regarding environment representations, both sparse and dense ones have been reported.
As a general understanding, mapping is the process to explore unknown regions of the surroundings. Therefore, as long as the mapping process can introduce additional information, the definition of maps can be multi-fold. Considering the topology of these representations, typical maps are categorized as Metric Map, Topological Map and Hybrid Map.

Metric maps use metric measurements as the basis of representation. The goal of metric mapping is a complete and precise modeling of the measurable environment in terms of Euclidean distance. Topological maps join locally salient features as Nodes and add Edges between pairs of nodes. Hybrid map is a fusion of local metric maps and global topological maps. In this thesis, topological environment models using omnidirectional camera are mainly considered.

0.4 Dissertation Outline and Contributions

0.4.1 Visual topological scene recognition

When topological modeling with imaging sensors is concerned, I consider the first requirement is to realize visual scene recognition. Furthermore, a simultaneous clustering and recognition fashion is preferred \cite{28}. This means that the topological scenes are to be automatically generated when the robot is moving in the environment, and to be recognized whenever the robot is back to a previously visited place.

The topological mapping and scene recognition techniques are efficient ways to model an environment with sparse information. When humans describe their surroundings, we normally use unique labels of the places such as “my office”, “the first part of the corridor” etc. It implies that humans have a mostly topological representation of their environment \cite{9}. It highly depends on their ability to learn ego positions based on visual hints. This intuitive observation can be extended to similar tasks for mobile robots. For mobile robots, the ability to visually detect scene changes and recognise existing places is essential. Moreover, since robots may have multiple tasks at the same time, these detecting and recognition methods are preferable with an online fashion and with minimum computational and memory cost in real-time. However, in most existing techniques for scene recognition, most of the computational time goes into feature extraction due to the complexity of the feature detector and descriptor, e.g. SIFT \cite{29}, SURF \cite{30}. Instead of computing complex, robust but computationally expensive descriptors, I would like to focus on robustly matching simple and lightweight descriptors, by forming the extraction and matching descriptors as a non-parametrical
probabilistic model.

**Part I** is conducted with environment modeling problem using omnidirectional camera as the only sensor. It mainly deals with SCAR problem using a color-based descriptor, the corresponding match algorithms and related scene recognition applications. It uses non-parametric methods to model SCAR problem and facilitate the calculation. After that, a novel lightweight scene recognition method is introduced. It uses an adaptive descriptor which is based on color features and geometric information for uncalibrated omnidirectional vision. The method enables robots to register new scenes (nodes) to a topological map automatically and solve the localization problem of mobile robot simultaneously in realtime. The descriptor of a scene is extracted in YUV color space and its dimension is adaptive depending on the segmentation result of the panoramic image. A DPMM is used to describe the perception process. The inference of the model is based on approximations of conditional probabilities of observations given the estimated models. It allows online inference of the mixture model in real-time (at 50Hz), which outperforms other existing approaches. A real experiment is carried out with a mobile robot equipped with an omnidirectional camera. The results show competence against the state-of-the-art.

The content of this part led to three conference publications \[31–33\], and one journal publication in IEEE Transactions on Robotics.

### 0.4.2 Visual topological navigation

The next question right after the scene recognition problem naturally is how to enable a mobile robot to move within an appearance-based topological map, namely the visual navigation problem. Recalling the traveling scenario shown in Figure 0.2.1, we found that the movement between each pair of stops is the essential problem to be solved. For visual control of a mobile robot, similar problem for such point-to-point navigation is named Visual Homing. Visual Homing enables a mobile robot to move to a reference (home) position using only visual information \[34–36\].

**Part II** extends the modeling problem to topological navigation using an omnidirectional camera. This part shows the applications of topological modeling in structured environments, which aims at maximizing the autonomy of mobile robots. It uses matched image key-points (e.g. SIFT) extracted from omnidirectional images as inputs to image-based visual servoing (IBVS) frameworks. First, I propose three visual homing methods based on scale, bearing and the combination of both. Second, considering the computational simplicity, I propose a simplified homing method which takes advantage
of the scale information of the keypoint features to compute a control law. Last but not least, navigation strategy using finite-state-machine is presented. Several technique details such as database managing, visual compass are discussed as well. The results of all these methods are compared both in simulation and experiments.

In topological visual navigation, the homing method is utilized to perform the transition between topological nodes. Compared to the methods based on metric maps [22][37][38], topological visual navigation framework has the following three major advantages:

1. **Sparse representation of environment** Usually the topological map used in topological visual navigation is created incrementally, by only considering feature changes. A typical representation of the environment is a collection of visual features at certain poses. The computational and memory cost are relatively low.

2. **Independent of precise maps** As a result, visual homing is less sensitive to error accumulation, which commonly occurs in metric mapping approaches.

3. **Lightweight planning** Path planning on metric maps can be computationally very expensive; in the contrary, the planning of visual topological navigation is based on graph structure with a relatively low cost.

   The primary challenge for Visual Homing problem is the estimation of the *homing vector*, which is defined as the direction in which the robot has to move along to reach the reference position. I tackle this problem by taking inspiration from the generic framework of Visual Servoing [39][40].

   In general, visual servoing approaches require the computation of the pseudo-inverse of a matrix whose size is proportional to the number of sampled features \( n \). To alleviate that, I propose an approach inspired from the visual servoing approach, but with a cost linear in the number of features. I will show that the resulting control law is stable, and its outperforming capability against the traditional visual servoing approach. The originality of the approaches presented in this chapter is that I take advantage of the scale information of the SIFT features into control. The observability of the controlled system is discussed in Chapter 6.3.

   Additionally, I discuss a quantitative evaluation of places to evaluate their suitability as reference positions. Although three landmarks are sufficient to determine a precise position on a plane, I found that the distribution of features will also affect the precision of homing due to the observation noise.
In this chapter, quantified quality factors based on the entropy of the feature distribution are proposed and I describe an experimental evaluation of the influence of this metric on the precision of the homing maneuver.

Tests conducted in real indoor environments and datasets confirm the outperforming performances and robustness of the proposed methods, leading to a complete real-time visual solution to topological mapping and navigation problem. A topological visual navigation framework centered on visual homing is introduced at the end.

The content of this chapter led to three conference publications [20, 41, 42], as well as one journal paper in IEEE Transactions on Robotics.

Please refer to the attached CV at the end of the thesis for more information regarding my academic record.
0. INTRODUCTION
Simultaneous Clustering and Recognition (SCAR) for multi-information fusion

A lightweight descriptor for omnidirectional camera is developed. It fuses vertical lines and color features.

A fast matching algorithm based on Dirichlet Process Mixture Model (DPMM)

Real-time online new node generation and recognition of existing nodes

Comparison with key-points based descriptors, such as SIFT

How strange a scene is this in which we are such shifting figures, pictures, shadows. The mystery of our existence—I have no faith in any attempted explanation of it. It is all a dark, unfathomed profound.

Rutherford Birchard Hayes (1822-1893)
Chapter 1

Information Fusion for Simultaneously Clustering and Recognition

1.1 Introduction

1.1.1 From perception to recognition

PERCEPTION is the process that converts raw sensor readings to expedient information. As we know, humans are good at perception. One important reason is that we use multiple sensors, such as eyes, nose and ears together, and gather information from different perspectives. Luo et al. in [43] provided an interesting biological explanation of multi sensor integration for animals. This perception process generally converts the raw information to several distinctive features. Taking the toy example shown in Figure 1.1.1, where we demonstrate the recognition process of “a can of cola”. We can intuitively imagine that it can be recognised by these two pieces of information: the shape of a can, and visual features (probably also by the taste, which we ignore in this specific discussion.) It also demonstrates that only by using one of the features may easily lead to false recognition. Conceptually, the recognition process can be summarised as shown in Figure 1.1.2 The objective of the task is usually to find the inherent label $\phi$, by taking into account several features $x^p$ obtained from the object. For this purpose, reliable descriptors of features are generally required.

1.1.2 From recognition to clustering

We must notice: the aforementioned scenario implies that the object must be learned beforehand. However, the bad news is that it is usually not feasible for robotic tasks. Instead, it is preferred to enable the robot to online learn everything from scratch. It is more equivalent to an integrated learning and recognition problem than a recognition problem alone. Such a problem can be instantiated as the example shown in Figure 1.1.3 It illustrates the
clustering and recognition problem of drinks. Given the large amount of selections, the system must fulfill the following two tasks. First, the drinks need to be clustered into groups, i.e. recognizing each type of drink among the selections. Second, given measurements observed from an object, the object can be labeled correctly regarding the type. We can see that clustering is the core algorithm in this scenario, whereas the complete knowledge of all the possibilities is necessary before the recognition process.
1.1.3 Simultaneous clustering and recognition

The problem becomes more interesting, if we can not access the complete knowledge. Considering robotic tasks are often carried out in the situation where none or only limited prior knowledge is accessible, e.g. robotic mapping of an unknown environment. Getting back to the toy example, instead of observing all the drinks before the recognition, we need to deal with the case that the drinks are shown in a arbitrarily-ordered sequence. The system is required to detect whether the current object has appeared in the past, if not, a new type of drink needs to be registered. I call such a process as Simultaneous Clustering and Recognition (SCAR) problem.

In this chapter, I introduce a generic framework for SCAR using a non-parametric Dirichlet hierarchical model. It enables online labeling, clustering and recognition of sequential data efficiently, while taking into account multiple types of sensor readings. The algorithm is data-driven, which does not depend on prior-knowledge of the data structure.

1.2 Information Fusion and Perception

1.2.1 Multi Information Fusion

In past decades, multi information fusion has shown its impact in different engineering fields, such as monitoring of complex structures, fault diagnosis and especially robotics. Most recent works treated multi information fusion problem in a decentralized fashion. In brief words, they first
considered multi information separately, reasoned/inferred them, then fused the conclusion of each sensor at the end. This pattern is potentially robust to failure of any one of sensors.

In order to elevate this, I introduce a generic framework which allows recognition tasks to take multiple sensor readings simultaneously. It is proved to be low cost in computation. Meanwhile, the sensor measures are coherently linked together via clustering. As a primary feature, the proposed algorithm uses non-parametric statistics to discover the inner relations among data from different subjects. It starts from zero prior-knowledge and takes sequence of concurrent data from different sensors as inputs. No specific training is required during the process. It enables the fused data to autonomously build new clusters and recognize an existing cluster in real-time.

The proposed model is stimulated by Dirichlet Process Mixture Model (DPMM) [46], which is nowadays widely used in texts classification and segmentation. The original algorithm takes only one type of input, such as words or letters. Moreover, the inference of a DPMM is computationally expensive, because sampling algorithms are usually required [47] for a large evident set. I extend the model to multiple observations from different sensors and develop an online approximation algorithm which enables fast inference in real-time.

The taxonomy of data fusion algorithms varies. We only list several related elements that are generally used in surveys.

**Decision Fusion**

Decision making is the most critical problem for intelligent systems. It is a general concept and is usually embedded into specific paradigms, such as failure detection, object recognition, pedestrian detection etc. Several work regarding decision fusion have been proposed in the scope of decentralized multi state representation. In [48], the authors introduced a decision fusion framework to fuse multi information by using confidence regions of the sensor model. Fauvel et al [49] used fuzzy set theory to fuse the decision from multiple classifiers. [50] introduced a force aggregation and classification model by fusing information from sensors with different resolutions.

**Sensory State**

The purpose of multi information fusion is to obtain information more robustly than using a single sensor. In most cases, the target information can be considered as goal states. At a early stage, Extended Kalman Filter (EKF) was widely used, where the perception outcomes from multi information are
taken as a unified state. This model is usually named as centralized state estimation. Several robotic applications have been proposed, such as [51] fuses vision and haptic sensor for object recognition; [52] used neural networks to fuse the sensor information of intelligent vehicles etc. However, these early works did not treat the multi information fusion mathematically efficiently. Moreover, the robustness to sensor failure is a big problem. Durrant-Whyte et al proposed a decentralized architecture named Decentralized Kalmann Filter (DKF) in [45], which handled multi information separately then fused the conclusions derived from each filter. The obvious advantage of DKF lies in its robustness to single sensor failure. Some recent researches still follow the same concept, such as object recognition by [53], segmentation problem by [54], pose estimation problem by [55, 56]. The proposed algorithm does not show explicitly decentralized characteristics. However, the joint probability given in Section 1.3 implies the independence of all sensor readings. It indicates that the confidence of each sensor is propagated to the posterior directly, which means sensor readings are not centralized as a single system state.

1.2.2 Clustering

In order to automate the classification and recognition process, an unsupervised learning algorithm is required. Sophisticated clustering algorithms usually depend on iterative calculation such as K-means, spectral clustering [57] or affinity-propagation [58]. A representative of online reasoning is chow-liu tree based segmentation [59] for static data and change point detection [60, 61] for sequential data. For extreme cases, the synchronization of multi information need to be taken care of [62] or spatial and temporal hints must be jointly considered [63]. In this chapter, an online naive change point detection algorithm is implemented, which is validated in Section 1.5.

1.2.3 Recognition and inference

Recognition is the core of most robotic applications. For example, robot topological mapping requires detection and recognition of loop-closure; semantic mapping usually requires recognition of objects; human-machine interfaces require recognition of human behaviors etc. Research targeting at these core problems attempts to seek the best algorithms to build efficient models which can represent this perception process efficiently.

Regarding inference approaches, hierarchical probabilistic methods based on statistical techniques achieved a great success in text mining and biological
information processing [64,65]. In this work, I alternate the classical mixture model to fit them with multiple types of observations. At the same time, it allows infinite increment of the number of labels. Furthermore, the model is to be learned, updated and inferred in real-time on-line.

In most of the related works, change-point detection [61,66,67] is the basis to segment a data sequence. In this work, as we are targeting at a lightweight method, the change-point detection is not feasible when using multiple hypothesis methods, such as particle filtering [61]. Instead, I use non-parametric statistic test to evaluate the labeling for each frame separately. This may cause instability in the output label. However, it reliefs the requirement of saving all the previous data of the sequence.

The theoretical advances in hierarchical probability frameworks, such as LDA [65] and HDP [64], provide a good support for the algorithm. The Dirichlet Process Mixture Model (DPMM) enables infinitely countable clusters for the measures, which can be used to represent the process of state recognition.

1.2.4 Assumptions and Contributions

Without loss of generality, the proposed algorithm deals with data with the following assumptions.

- The multi information are synchronized, or they can be treated as a complete observation unit if they have different sampling rates;
- Features of the sensor readings are observable and computationally feasible in near real-time;
- As a requirement of DPMM, multi information in the set must be exchangeable, which indicates that the labeling of a reading does not depend on whether such readings appear earlier or later.

The objectives that I want to achieve in this chapter are twofold:

- Modeling multi information process using hierarchical probability model. The model of the recognition process depends on parameter set with small cardinality;
- A concise approach for on-line inference of the proposed Dirichlet Process Mixture Model.

The remainder of this chapter is organized as follows. Inspired by the traditional Dirichlet Mixture Model, I start with proposing the hierarchical model for online recognition using multi information. The full inference of the model will also be introduced. Then I introduce an approximate method for fast inference of the model. The evaluation of the model using simulation
1.3 Model Formulation

1.3.1 Model of Information Fusion

Recalling the toy example discussed in the introduction, each of the sequential observations $x$ can be considered as a sample drawn from a base distribution. The distribution is determined by the potential label $\phi$, which is a sample drawn from a discrete space $G$. The model is depicted as shown in Figure 1.3.1. $G$ is a Dirichlet process distributed with a base distribution $H$ and a concentration parameter $\alpha$. The base distribution is the mean of the DP and a concentration parameter $\alpha$ is as an inverse variance. The distribution $G$ itself has point masses, and the draws from $G$ will be repeated by sequential draws considering the case of an infinite sequence. Additionally, $\phi_t$ is an indicator of the cluster identity, to which the current data set at time $t$ belongs. In order to help the readers to understand this representation, I carry out minor discussion here.

**Chinese Restaurant Process (CRP)**

Since the process can be considered as a partition problem, a CRP model is usually used. In probability theory, the CRP is a discrete-time stochastic
process, whose value at any positive-integer time $n$ is a partition $B_n$ of the set \{1, 2, 3, \ldots, n\} whose probability distribution is determined as follows. At time $n = 1$, the trivial partition \{\{1\}\} is obtained with probability 1. At time $n + 1$ the element $n + 1$ is either:

- added to one of the blocks of the partition $B_n$, where each block is chosen with probability $\frac{|b|}{n+1}$ where $|b|$ is the size of the block, or
- added to the partition $B_n$ as a new singleton block, with probability $\frac{\alpha}{n+1}$. \[1\]

Intuitively, the process can be illustrated as Figure 1.3.2. It represents a restaurant scenario as follows. A new customer enters a restaurant, the chance that (s)he picks a specific table is proportional to the number of customers already sit in front of the table. Besides, the chance to choose a new table, is proportional to the concentration parameter $\alpha$. When we consider each table represents a specific set of parameters, such as those defined the cluster of measurements, the CRP provides a prior distribution of the newly arrival measurement. In this case the base distribution $H$ is simply $\{1\}$. Meanwhile, we could also see that a greater $\alpha$ leads to higher chance to start a new cluster, verse vesa.

\[\begin{array}{|c|c|c|c|c|}
\hline
\text{Case} & P(\bullet = 1) & P(\bullet = 2) & P(\bullet = 3) & P(\bullet = \?) \\
\hline
\alpha = 1 & \frac{4}{9+1} & \frac{2}{9+1} & \frac{3}{9+3} & \frac{4}{9+4} \\
\alpha = 2 & \frac{2}{9+2} & \frac{2}{9+2} & \frac{2}{9+2} & \frac{2}{9+2} \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
\alpha = \alpha & \frac{\alpha}{9+\alpha} & \frac{2}{9+\alpha} & \frac{3}{9+\alpha} & \frac{\alpha}{9+\alpha} \\
\hline
\end{array}\]

Figure 1.3.2: Chinese Restaurant Process

\[1\text{In the traditional scheme, } \alpha \geq 1.\]
Information fusion model

For the case of multiple information, following the notations in Figure 1.3.1, I design a DPMM as shown in Figure 1.3.3, where the parameters are depicted in rectangles, and random variables are in circles. It utilizes the fact that one measurement can be considered as a sample from both the distribution defined by the label or from $p$ independent sensors. The consisted components are explained in the following subsections.

$$
\begin{align*}
\phi_t & \sim N(x_t, \alpha) \\
n & \sim 1:N \text{ Data} \\
k & \sim 1:K \text{ Clusters} \\
p & \sim 1:P \text{ Sensors}
\end{align*}
$$

**Figure 1.3.3:** Directed Acyclic Graph (DAG) of the proposed model for recognition by multi information fusion

According to the previous discussion, $\phi_t$ is the target variable of inference. By integrating over $G$, the drawing of $\phi_t$’s can be depicted as a CRP:

$$
\phi_t \mid \phi_{1:t-1} \sim \frac{\sum_{n=1}^{t-1} \delta_{\phi_n} + \alpha H}{t - 1 + \alpha}
$$

where $\delta_{\phi_n}$ is an indicator of a certain frame $n$ which is labeled as $\phi_n$, i.e. a mass point function locates at $\phi_n$. We must notice that this assumption implies that the more we see a certain cluster of data, the higher a prior that
data from such cluster may be observed again. The target problem is then converted to an estimation of

\[ P(\phi_t \mid \phi_{\setminus t}, G, x; \theta; \alpha, \beta) \]

where \( \phi_{\setminus t} \) is the full set of indicators excluding the current one at stamp \( t \), namely the history of labels. Sets of random variables and parameters are shown in bold.

### 1.3.2 Multi Information Perception

The multi information of \( P \) different types of readings are modeled as the orange plate (encircled by dashed lines) shown in Figure 1.3.3. For all \( N \) readings in the sequence, \( x^p_t \) represents the perceived information acquired at time-stamp \( t \) from sensor \( p \). Taking discretized readings as an example, perceived information from raw sensor data can be represented as histograms \[32\]. Assuming there are \( K \) different clusters, \( \theta_k \) is a matrix of dimension \( K \times Z^p \), where \( Z^p \) is the number of possible histograms for sensor \( p \). \( x_t \)'s of dimension \( Z^p \) are drawn from \( \theta_k \). \( Z^p \) is the cardinality of readings for sensor \( p \).

On one hand, \( x^p_t \) is inherently determined by its label \( \phi_t \); on the other hand, we can also consider the sensor readings as samples drawn from a sensor model \( \theta^p_k \) for cluster \( k \), with a sensor model prior \( \beta^k \). So far, we build a model of two sub-processes, namely the sensing process and perception process, which serves as a basis to build data-driven inference model for the recognition problem.

### 1.3.3 Model Inference

As a summary of the proposed model,

\[ G \sim Dir(\alpha H) \]

\[ \phi_t \mid G \sim G \]

\[ x^p_t \sim F(\phi_t, \theta^p_{\phi_t}) \]

\( F \) represents the generation function of the measurements from the base models, regarding label \( \phi_t \). The joint probability can be written directly as,

\[ p(\phi G \theta x; \beta) = \prod_{p=1}^{P} \prod_{k=1}^{K} p(\theta^p_k; \beta^p) \prod_{t=1}^{N} p(G \mid H, \alpha) p(\phi_t \mid G) \prod_{p=1}^{P} p(x^p_t \mid \theta^p_{\phi_t}) \]
In order to factorize it to independent components, we can integrate the joint probability over $\theta_1, \theta_2 \ldots \theta^P$ and $G$,

$$p(\phi \mathbf{x}; \beta) = \int_{\theta_1} \cdots \int_{\theta^P} \int_G p(\phi \ G \ \theta \ \mathbf{x}; \beta) \ dG \ d\theta_1 \ldots d\theta^P$$

$$= \int_{\theta_1} \prod_{r=1}^K p(\theta^r_1; \beta^1) \prod_{t=1}^N p(x^r_t \mid \theta^r_\phi_t) \ d\theta^1$$

$$\ldots$$

$$\times \int_{\theta^P} \prod_{r=1}^K p(\theta^r_P; \beta^P) \prod_{t=1}^N p(x^r_t \mid \theta^r_{\phi_t}) \ d\theta^P$$

$$\times \int_G \int_H \prod_{t=1}^N p(\phi_t \mid G)p(G; H\alpha) \ dH \ dG \tag{1.1}$$

The last component is an expectation given $G$, i.e. $E_G [p(\phi_1 \phi_2 \phi_3 \phi_4 \ldots \phi_N \mid G)]$. According to the features of the Dirichlet process, it is proportional to the product $\prod_{t=1}^N p(\phi_t \mid \phi \backslash_t) \propto p(\phi_t \mid \phi \backslash_t)$. Therefore,

$$\int_G \int_H \prod_{t=1}^N p(\phi_t \mid G)p(G; H\alpha) \ dH \ dG \propto \sum_{t=1}^{N-1} \delta_{\phi_t} + \alpha \delta_{\bar{k}} \tag{1.2}$$

where $\delta_{\phi_n}$ is a mass point function located at $\phi_n$. $\bar{k}$ is the indicator for a new cluster.

The components in the front can be treated in a similar manner. Take the integral of $\theta^P$ for instance, using $n^k_v$ to represent the number of measures which is the $v$-th element in $\theta^P$ within cluster $k$.

$$\int_{\theta^P} \prod_{k=1}^K \prod_{v=1}^Z p(\theta^P_k; \beta^P) \prod_{t=1}^N p(w_t \mid \theta_{\phi_t}) \ d\theta^P$$

$$= \prod_{k=1}^K \int_{\theta^P_k} \frac{\Gamma(\sum_{v=1}^Z \beta^P_v) \Gamma_{\theta^P_k}}{\prod_{v=1}^Z \Gamma(\beta^P_v)} \prod_{v=1}^Z \theta^P_{\theta^P_k \beta^P_v} \prod_{v=1}^Z \theta^P_{\theta^P_k \beta^P_v - 1} \ d\theta^P_{\theta^P_k} \tag{1.3}$$

$$= \prod_{k=1}^K \int_{\theta^P_k} \frac{\Gamma(\sum_{v=1}^Z \beta^P_v + n^k_v)}{\prod_{v=1}^Z \Gamma(\beta^P_v + n^k_v)} \prod_{v=1}^Z \theta^P_{\theta^P_k \beta^P_v + n^k_v} \prod_{v=1}^Z \theta^P_{\theta^P_k \beta^P_v + n^k_v - 1} \ d\theta^P_{\theta^P_k}$$

since from the integral of Dirichlet distribution,

$$\int_{\theta^P_k} \frac{\Gamma(\sum_{v=1}^Z \beta^P_v + n^k_v)}{\prod_{v=1}^Z \Gamma(\beta^P_v + n^k_v)} \prod_{v=1}^Z \theta^P_{\theta^P_k \beta^P_v + n^k_v - 1} \ d\theta^P_{\theta^P_k} = 1 \tag{1.4}$$
The joint probability is represented as follows.

\[ p(\phi \mid x; \beta) \propto \prod_{p=1}^{P} \prod_{k=1}^{K} \frac{\Gamma(\sum_{v=1}^{Z^p} \beta^p_{v}) \prod_{v=1}^{Z^p} \Gamma(\beta^p_{v} + n^k_v)}{\prod_{v=1}^{Z^p} \Gamma(\sum_{v=1}^{Z^p} \beta^p_{v} + n^k_v)} \times \left( \frac{\sum_{t=1}^{N-1} \delta_{\phi_t} + \alpha \delta_{\phi_{\phi_t}}}{N - 1 + \alpha} \right) \]

(1.5)

When we consider a collapsed Gibbs sampling process on the cluster indicator \( \phi_t \) at time \( t \), we have

\[ p(\phi_t \mid \phi_{\setminus t} \mid x; \beta) \propto p(\phi_t \mid \phi_{\setminus t} \mid x; \beta) \]

(1.6)

However, the huge size of \( Z^p \) makes the direct inference not possible. Usually sampling methods [47] are used to estimate the posterior. Nevertheless, the sampling based algorithm is usually computational expensive as well. It is required to find an online approximation algorithm, in order to make the algorithm work in real-time.

### 1.4 Approximation

In this section, I introduce the approximation algorithm to infer the proposed DPMM. For the case where measurement \( \phi_t = k \), for simplicity, I rewrite equation (1.5) as follows.

\[ p(\phi_t = k \mid \phi_{\setminus t} \mid x) \]

\[ \propto \prod_{p=1}^{P} \frac{\Gamma(\sum_{v=1}^{Z^p} \beta^p_{v} + n^k_v)}{\Gamma(\sum_{v=1}^{Z^p} \beta^p_{v})} \left( \frac{\sum_{t=1}^{N-1} \delta_{k} + \alpha \delta_{\phi_{\phi_t}}}{N - 1 + \alpha} \right) \]

(1.7)

We could see from equation 1.7 that the first \( P \) components \( \xi^p() \)'s calculate the gamma function of the count of a certain observation over all possibilities. In other words, they represent the probability of a certain measure showing up in an observation sequence. Therefore, it can also be considered as a measure of the similarity of the current observation to all the predefined models. As a result, we don’t need sampling methods to estimate this posterior if we can approximate the underlying similarity between current observation and reference models. This conclusion leads to very flexible solutions to recognition problems, since the similarity between observation and model can be obtained by various criteria, e.g. the number of matched key-point
features, result of spectrum analysis, dot product of observation vectors etc. In the end, a scalar will be used to indicate this similarity. The resulting scalar \( s \) can further represent the observation as a sample from a distribution of exponential family, such as zero-mean Gaussian distributions [e.g. \( C \cdot e^{-s^2} \)] or Beta distribution [e.g. \( Be(1, S) \) where \( S > 1 \)].

However, another factor must be considered. It is the weighting factor among all sensors. As for equation 1.5, this factor is modeled by prior \( \beta \). Joining with the approximation by exponential family distribution,

\[
\xi^p(x_t^p \mid \theta_k^p) \equiv e^{-(\omega_p s^2(x_t^p, \hat{\theta}_k^p))}
\]

A set of weights for sensors can be used as follows.

\[
\begin{aligned}
p(\phi_t = k \mid \phi \setminus x) \propto \left( \frac{\sum_{t=1}^{N-1} \delta_k + \alpha \delta_{\phi_k}}{N - 1 + \alpha} \right) e^{-\sum_{p=1}^{P} \omega_p s^2(x_t^p, \hat{\theta}_k^p)}
\end{aligned}
\]

where \( \hat{\theta}_k^p \) is the incrementally estimated model, and \( s() \) depicts the matching result between the current observation and the model. One example of the incremental estimation of the model is given in section V.B of [32].

### 1.5 Simulation

The simulation with multi inputs for online clustering and recognition is introduced in this chapter. I simulate three synchronized sensor readings, which are observed from a system with changing states. The ground truth of the changing state is shown in the first block of Figure 1.5.1. Subplot A, B and C show the readings from three different sensors. Please note that sensor reading C provides only noisy signal, independently to the state change. I use the case C to simulate that low information sensor readings could be successfully omitted by the proposed decentralized framework. The sensor models are zero-biased Gaussian distributions. We can first check the likelihoods of the joint probability regarding state 1, 2 and 3 without considering change-point detection, in order to validate the distinctness drawn from equation 1.7. The joint likelihood for three sensors is shown in figure 1.5.2. It shows that if the cluster of the data is given, the model could validate sensor readings as samples from each cluster. The remaining problem is that the clusters of date (respecting each state of the system) need
to be automatically detected and incrementally generated. To this end, I use a naive change-point detection algorithm, since the noise level of the simulated data is low. A time-stamp is considered as change-point when the posterior of the observations is lower than a threshold in conjugated 5 readings. For sophisticated change-point detection algorithm such as particle filter, please refer to [61, 67, 68]. The result is shown in Figure 1.5.3. The first subplot shows the posterior of MAP (Maximize-a-Posterior) result. The change-point detection is shown in the second part. At the end, I present the resulting labeling (in blue) against the ground truth (in red).

The results indicate that the proposed DPMM model is able to detect and register new clusters of data online, while performing recognition task simultaneously.
1.6 Discussion and Summary

We observe from equation 1.8 that the inference of the DPMM model falls back to a product of the likelihood of each sensor reading and a CRP process. It shows that given the observation $x_p^T$, all the sensor perception model are independent. This is consistent with the original model design of Figure 1.3.3. Therefore the system state can be easily written as a decentralized way. It means that a DKF filter is also applicable as post-processing.

A recognition algorithm usually leads to a big set of parameters. The choice of parameters especially thresholds will lead to dramatically change in the final result. Equation 1.8 shows that the proposed algorithm depends on the weighting factors for sensors and prior of the CRP process. The influence of prior $\alpha$ for the CRP process can be ignored when the number of measurements $N$ goes large. The weighting factors can either be chosen empirically (e.g. vision usually plays a more important role in object recognition than laser) or enable them to be adaptively tuned by variance analysis (e.g. greater variance among different measures leads to higher weight) etc.

In this chapter, I presented DP-Fusion, an on-line information fusion framework for multi information based on Dirichlet Process Mixture Model. It combines synchronized sensor readings to automatically cluster data into models, while recognizing data from existing models simultaneously. Results showed its advantage of on-line computing mode and low computational cost. This study also implied that the inference of a DPMM can be approximated by the product of the conditional probability. We can envision that similar concept can be borrowed to solve other inference problem as well. This model is adopted for the inference of scene recognition problem in the chapter 3.
Figure 1.5.3: Simulation of change point detection and clustering result
Sophisticated modeling for [SCAR] problem using [DPMM] has been discussed in the previous chapter. The next question is how to design a proper descriptor for the target scene recognition problem, then combine the descriptor with the proposed [SCAR] model. Fast adaptive color tag (FACT) descriptor for uncalibrated omnidirectional vision is presented in this chapter. It enables the mobile robot to recognize scenes using image appearances and autonomously add nodes into a topological map while fitting real-time requirements.

2.1 Scene Recognition with an Omnidirectional Camera

Several techniques are used to describe the surrounding environment of a robot. One of the major differences lies in the various descriptors used by structure reconstruction. We could see that many algorithms utilize keypoint-based features on perspective cameras, e.g. PTAM [23] used mainly FAST corners [69]; FAB-MAP [26] used mainly SIFT [29] or SURF [30]. However, not many applications or descriptors have been reported on omnidirectional camera, such as the example depicted in Figure 2.1.1. The main reason is the distortion introduced by the nonlinear transformation from the mirror shape. The nonuniform resolution will greatly affect the stabilities of patch based descriptors.

2.1.1 Scene descriptors

Most existing place recognition systems assume a finite set of place labels. And the task is to classify the labels for each image frame. These classifier-based approaches [70] are limited with the applications in pre-defined or known environments. One of the mainstream techniques for visually scene recognitions is based on object detection [71–74]. A representative scenario of these methods is to first detect known objects in the scene, then maximize
the posterior of the place label given these recognised objects. Methods based on similar concept \[75, 76\] used keypoint-based features for complete scenes. These methods are very robust when the objects are correctly detected. Nevertheless, the state-of-art object detection methods \[29, 30\] are usually computational expensive. They could be easily unfeasible on computers with limited resources, even with optimizations \[77, 78\], let alone the robot may have simultaneous tasks other than place recognition.

Several lightweight keypoint descriptors were developed \[79, 80\] as well and got widely applied in scene recognition problems \[81–83\]. Unfortunately, most applications only deal with either categorization of finite number of known places or only available for off-line inferences.

Beside the keypoint-based approaches, descriptors using the transformation/inference of whole images \[84–89\] are also popular. Specifically, for omnidirectional cameras, vertical lines are widely used. Scaramuzza et al proposed a simple descriptor using histogram extracted from three circles along the radial direction \[90\]. Our previous work optimized this descriptor for more flexible parametrization \[91\]. The most similar to the proposed FACT descriptor is the fingerprint of a place \[92, 93\]. Both fingerprint and FACT use segments from unwrapped panoramic images. The difference is that both \[92\] and \[93\] used laser range finder to help the matching of the descriptors, and FACT used only color information from the segments.

As for color features, beside the fingerprint of place \[92\], a detailed report on the state-of-art can be found in \[94\]. Generally speaking, color feature is a weak descriptor, as it can be affected by lighting conditions easily. It is the main reason why I need to use a statistical method in this work to minimize the uncertainty. Fei-fei et al \[95\] proposed a keypoint-based approach using this framework to cluster natural scenes. Nevertheless, the proposed work deals with fusion of different information rather than sole keypoints.
2.2 Formation of FACT

2.2.1 Unwrapping of Omnidirectional Image

An unwrapped image will facilitate the extraction of major vertical lines, since all the radial lines are projected into vertical direction. Hough Circle Detection algorithm is firstly performed in order to obtain the radius of effective FOV and the center coordinate. The detection results are shown as Figure 2.2.1 The outermost circle is taken as the effective FOV since its inner part covers all valid information of the panoramic image. The estimated image center is taken by the circle center shown in Figure 2.2.1(c).

![Figure 2.2.1: All the circles detected in the raw image. The outermost circle is extracted.](image)

The raw omnidirectional image is then unwrapped using interpolation as shown in Figure 2.2.2 The unwrapping mapping makes the detection of vertical lines more straightforward.

2.2.2 Segmentation of the Panorama

Since this work proposes a descriptor based on color features and segmentation on the panoramic image, the robustness of the method for detecting vertical edges is one of the key problems to solve. The segmentation is made by extending the dominant vertical lines in the panoramic image as depicted in Figures 2.2.2, 2.2.3, and 2.2.4. The technical details are explained as follows. After unwrapping the raw panoramic image (Figure 2.2.2), I apply in sequence Sobel filtering (only along the $x$ direction), Otsu thresholding, and morphological operators to extract the most dominant vertical lines.

---

1 The panoramic is with resolution 1024x176 in the tests.
Figure 2.2.3 shows the result of vertical extraction process. Note that only half of the unwrapped image is shown here because of the width limitation.

![Figure 2.2.2: A unwrapped result](image1)

![Figure 2.2.3: An output of vertical edges detection](image2)

![Figure 2.2.4: The segmentation result](image3)

The dominant vertical lines are chosen based on their length. The lines with length above the median are retained. Morphological operators are used just to fuse those lines, which are too close to each other, into a single line. As a summary, the detailed processing phases are shown in Figure 2.2.5.

As observed in Figure 2.2.4, the vertical lines partition the panoramic image into multiple regions. In the next section, I will explain how to extract the descriptor from these regions.

### 2.2.3 FACT Descriptor and Topological mapping

In this subsection, I describe the components that construct a FACT descriptor. The color representation in RGB color space is not suitable, since it is sensitive to illumination changes which may be caused by translation and rotation of the omnidirectional camera, as well as different times of the day. Alternatively, we use the YUV color space. The euclidean distance between
The segmentation process two color sites is shown in equation 2.1

\[
\begin{align*}
U_i &= (0.7 \times R_i - 0.6 \times G_i - 0.1 \times B_i) \\
V_i &= (0.9 \times B_i - 0.3 \times R_i - 0.6 \times G_i) \\
Dis_{1,2} &= \sqrt{(U_1 - U_2)^2 + (V_1 - V_2)^2}
\end{align*}
\]  

(2.1)

Figure 2.2.5: The segmentation process

2.2.4 Construction of FACT Descriptor

The descriptor was extracted based on the segmented unwrapped image explained in the previous section. For each region between two vertical lines, the average color value in the U-V space is extracted. Comparing to other keypoint-based or edge-based descriptor, an obvious advantage in the proposed approach is that the similarity between features in the U-V space will be simply measured in terms of a 2D Euclidean Distance. The descriptor
is formed by the U-V color information and the width $W$ (in pixels) of the region, which is delimited between two vertical edges. Instead of taking each pixel in every region into account, I directly use the average of U-V value that is calculated for each region. $(U_i, V_i)$ indicates the color information of region $i$.

One primitive idea is that even if the width of each region may change during the translation of the camera, the projected area in the real-world can be well-determined in a local neighborhood, as long as the segmentation stays consistent. In this case, the average value for a certain region in color space will keep constant. On the other hand, we must avoid the false positive caused by color similarity of regions. For example, the difference between a green cup and a green cabinet may be very small in color space, but the geometric features of these two are distinguishable. Therefore, I employ the width of correspond region $W_i$ as the third dimension of the descriptor. By testing the ratio of the width of corresponding regions, the descriptor can get more reliable results. If let $N$ be the number of regions segmented from the unwrapped image, the dimension of the FACT descriptor of a scene is $3 \times N$.

A sample descriptor $D$ is shown in Eq. (1). Each column in the descriptor is named as a Tag.

$$D = \begin{pmatrix} U_1 & U_2 & \cdots & U_N \\ V_1 & V_2 & \cdots & V_N \\ W_1 & W_2 & \cdots & W_N \end{pmatrix} \quad (2.2)$$

### 2.3 Naive Matching algorithm

The matching stage is the fundamental part of the method, as color descriptors are very weak. As a part of comparison, I first demonstrate a naive matching algorithm using a 3-phase strategy to compare two descriptors $D_1$ and $D_2$ as follows,

1. **Test 1: Tag Matching in the U-V Color Space**
   
   As shown in Figure 2.3.1, the distance in the U-V color space is first calculated on each region in the current image using the 2D Euclidean Distance. The minimum of the distance array $Dis_i$ is selected as represent of the region. If $Dis_i$ is smaller than a pre-defined threshold, $TH_{local}^3$, this region $i$ is considered matching with the corresponding region in the query image.

---

2 According to experiment results, $N$ is usually between 20 and 100 for typical office environments.

3 We choose $TH_{local} = 3e^{-4}$ in this work, and I used a $10 \times 10$ U-V color space. According to tests, this threshold applies very strict filtering of false positive.
region in $D_1$ and passes Test 1.

2. **Test 2: Tag matching in geometric space**
   If the Tag passes the first test, the width of the region is then examined. Here the comparison is made in terms of the ratio between the widths of the two regions. For instance, if region 1 in the current image matches with region 4 of another image according to test 1, then the ratio between the width of region 1 in $D_2$ (namely $W_{2,1}$) and the width of region 4 in $D_1$ (namely $W_{1,4}$) must satisfy the inequality $\frac{1}{3} < \frac{W_{2,1}}{W_{1,4}} < 3$. The range of this ratio has been found empirically. If this test can pass, the corresponding region in the database will be eliminated when matching with other regions in the current image.

3. **Test 3: Descriptor matching**
   Tests 1 and 2 are executed recursively until all the Tags in $D_2$ have been tested. The final score of current image is given by the ratio between the number of matching regions in the database and length of $D_1$, namely $j$.

The matching process is summarized in Algorithm 1. It leads to a Tag-level comparison, as shown in Figure 2.3.1

![Figure 2.3.1: A schematic diagram of Test 1 during the matching process.](image-url)

**Pair-wise Distance of Color Features**

In order to test the saliency of the proposed lightweight color feature, i.e. the first two dimensions in the UVY-color space, it is necessary to cross compare the belief that frame $n$ belongs to the topological node defined by frame $m$. 

39
Algorithm 1: Naïve Nodes Matching and Node Identification

Input:
- Input panorama image: $Im(n)$
- Existing Node and FACT tags: $Featurebase(m)$
- Distance threshold: $TH_{global} = 78\%$
- Matching local threshold: $TH_{local}$

Output:
- Matching Distance: $Distance(n)$
- Feature for current frame: $Feature(n)$
- list of nodes: $Nodelist$
- list of FACTs: $FACTlist$

1. if $Nodelist$ is empty, initial state then
   2. for each region $i$ do
   3.     Extract FACT Tags $\rightarrow Featurebase(1)$;
   4.     $1 \rightarrow Nodelist, Tags \rightarrow FACTlist$;

5. Extract FACT Tags $\rightarrow Feature(n)$;
6. for each Node $m$ in $Nodelist$ do
7.     for each Tag $k$ in $Feature(n)$ do
8.         for each Tag $j$ in $Featurebase(m)$ do
9.             if $n$ matches the feature in $j$ then
10.                $Distance(k)(j) = \sqrt{\Delta U^2 + \Delta V^2}$;
11.                $Distance(k) = \text{Min}(Distance(k)(j))$;
12.                $s = \text{argmin}_{s \in j} Distance$;
13.                if $Distance(k) < TH_{local}$ and the width fit the geometric constraint and $s$ is not in the current matched list then
14.                    Add $s$ to the matched list;

15. $Bel(n|Nodelist) = \frac{\text{length of the matched list}}{\text{dimension of Featurebase(m)}}$;
16. if $Bel(n|Nodelist) < TH_{global}$ then
17.     Update $Nodelist$ and $FACTlist$;
18.     $m++$;
19. Goto line 5 until pre-defined condition;
The pair-wise beliefs in the sequence, indicating line 11 in Alg. 1, construct a distance matrix as shown in Figure 2.3.2 using the dataset described in section 2.5. It intuitively shows the distinctions among different frames using color-based appearances. We can see that the adjacent frames from the image sequences show higher beliefs to be clustered together. At the same time, the lighter color blobs indicate that there are certain possibilities for those non-adjacent frames to be classified as the same scene as well, in the case that the robot may have returned to a previously visited place.

![Distance Matrix based on Euclidean distance in UV Space](image)

**Figure 2.3.2:** Distance matrix for the pair-wise beliefs

**Refinement of Result**

By using the naive algorithm, quite a few redundant nodes may be created. The reason is that these nodes are generated because some important vertical edges are occluded by obstacles or moving people when relatively low number of Tags are detected. Therefore, I perform an off-line method for refining the Nodelist in the reverse sequence as a complement. This refining process is similar to the matching process, except that we use a lower threshold $TH_{\text{refine}}$ (empirically $TH_{\text{refine}} = 70\%$) to enable the tolerance of belief. It
starts from the last node in the Nodelist and compare it with all the previous nodes using algorithm [1]. If some node(s) can be matched with others in the Nodelist, they will be fused. The process runs repeatedly until it reaches the base node Node1. The list of all the remaining valid nodes, along with the refined FACTlist, is the output of the method.

2.3.1 Test of robustness in dynamic environment

The method described in this work is supposed to omit the disturbance when less than 30% of the panoramic image in horizontal direction is affected. This characteristic is especially valuable when people may be walking around during the motion of the robot, after the topological map is trained. Ideally speaking, if more than 30% is covered by dynamic objects or people, the robot will take the current scene as an unknown area. Take Figure 2.3.3 for instance, the people outlined by green is not supposed to influence the recognition result comparing with the left image taken at the corridor. The important fact should notice that the matching result of image is based on the number of regions, not the matching area. This changes when dynamic objects/people occludes the vertical edge. Therefore, the experiment based on how much ratio of the image is occluded is meaningless, because the ability of recognition depends on the different types of environments. Experiments are designed in a more practical way by having several people walking around in the field of view of the camera. The experiment for the robustness test is designed as below:

1. Locate the robot in a typical indoor environment, and generate only one node for the test environment;
2. A number of people randomly walk within the field of view and take the log separately;
3. Change to another scene and repeat from step 1.

The experiments were taken in real-time and the test results are given in figure 2.3.4. The sub-figure (a)-(d) show the test results in an office room, in a corridor, by stairway, and in a coffee room respectively. Because of the different color textures of these indoor environments, the robustness of FACT appears differently. For the office environment, three people walked around consequently and they crowded together time after time. So the result shown in (a) shows that when people occlude intolerable regions for the algorithm, the result goes below the threshold. The test result (b) and (c) show the same trend. Most images grabbed at the stairway can have high score, because the texture at the test point is very diversified and the image has very distinguishable regions in color. Another reason is that the
space is larger than that in the coffee room or office room, so during the test, people could walk farther compared with that in other places. Therefore, sometimes only a small part of the FOV is occluded, which also makes the result look better. The Figure 2.3.4(e) shows a result of an experiment on sensitivity, in which we suddenly cover the whole FOV and the algorithm shows its sensitivity to the sudden change, while maintaining the frame rate around 20fps.

2.3.2 Drawbacks of Naive Matching algorithm

The naive matching algorithm in Alg. 1 has been studied and compared from different perspectives [20, 97-100]. According to further study, the major disadvantages of the original approach are as below:

- The matching step is a point estimator, without considering probability and multi hypotheses.
- The false positive ratio of scene changing detection is high, therefore it required an off-line refinement.
- The parameter-set is big. Five parameters need to be adjusted in total.

Considering these shortcomings, we refactorize it as a probability-based framework as following. An alternative DPMM is specified for topological mapping in the next section.
Figure 2.3.4: The test result of the robustness of FACT. The red line indicates the threshold $TH_{global}$ which defines the matching status of current image.
Chapter 3
DP-FACT: Scene recognition with DPMM

3.1 Modeling in term of SCAR

We can consider scene recognition as a SCAR problem, by using visual features. It is a process of detecting changes and re-localizing in an existing topological environment model. By taking the results of the model introduced in chapter 1 and the novel descriptor in chapter 2, the model is shown in Figure 3.1.1 which consists of two measurements, i.e. the color information and geometrical width embedded in Fast Adaptive Color Tags (FACT). The parameters are depicted in rectangles, and random variables are in circles.

![System Model](image)

Following the explanations in chapter 1, $G$ is a Dirichlet process distributed with base distribution $H$ and concentration parameter $\alpha$. The base
distribution is the mean of the DP and the concentration parameter $\alpha$ acts as an inverse variance. The distribution $G$ itself has point masses, and the samples from $G$ will be repeated by sequential samples considering the case of an infinite sequence. Additionally, $\phi$ is an indicator of which cluster does the current image at time $t$ belongs to.

The observable random variables from the model are two multinomial distributions $g_t$ and $w_t$, which represent two histograms by accumulating the number of features which hit their own discretized space. Taking $w_t$ for an instance, it is a multinomial distribution that represents a single histogram of different width of Tags in one DP-FACT feature. The dimensions of a sample $g_t$ and $w_t$ are $D_{uv}$ and $D_w$ respectively, indicating the dimensions of the discrete UV space and width space. The number of samples is represented by $N$, which is equal to the number of sequential frames during one experiment.

By only considering $w_t$ for example, as it is a multinomial distribution, $w_t$ is subject to a Dirichlet distribution prior $\omega_j$. Assuming there are $K$ different scenes, $\omega$ will be a matrix of $K \times Z$. $w_t$'s of dimension $Z$ are drawn from $\omega$. $Z$ is the number of possible histograms given the maximum number of Tags of a frame, which is a large number. Since we use an approximation method for the inference in Section 3.2, the precise expression of $Z$ is not necessary. Please notice that because $\theta$ and $\omega$ are discrete, $P(\theta_{t1} = \theta_{t2}) \neq 0, P(\omega_{t1} = \omega_{t2}) \neq 0$, for different time stamps $t1$ and $t2$. In summary,

\[
\begin{align*}
G &\sim \text{Dir}(\alpha H) \\
\phi_t | G &\sim G \\
g_t &\sim F(\phi_t, \theta_{\phi_t}) \\
w_t &\sim Q(\phi_t, \omega_{\phi_t})
\end{align*}
\]

$F$ and $Q$ represent the generation functions of the measurements from the parametric models, regarding the label $\phi_t$. By considering a standard stick-breaking construction process [101], and integrating over $G$, the sampling of $\phi_t$'s is represented as:

\[
\phi_t | \phi_{1:t-1} \sim \sum_{n=1}^{t-1} \delta_{\phi_n} + \frac{\alpha H}{t - 1 + \alpha}
\]

where $\delta_{\phi_n}$ is an indicator of a certain frame $n$, which is labeled as $\phi_n$, i.e. a mass point function locates at $\phi_n$. The target problem is then converted to an estimation of $P(\phi_t | \phi_{\setminus t}, G, g, w, \omega, \theta; \beta, \lambda)$, where $\phi_{\setminus t}$ is the full set of indicators excluding the current one, namely the historical labeling.

Adopting equation (1.7) from chapter 1, the representation of the label posterior can be obtained as shown in equation (3.1).
\[
p(\phi_t = k \mid \phi_{\setminus t} g w) \\
\propto \frac{\prod_{v=1}^{Z} \Gamma(\lambda_p + n_{p}^{k}) \prod_{v=1}^{Y} \Gamma(\beta_q + c_{q}^{k})}{\Gamma(\sum_{v=1}^{Z} \lambda_v + n_{v}^{k}) \Gamma(\sum_{u=1}^{Y} \beta_u + c_{u}^{k})} \left( \frac{\sum_{t=1}^{N-1} \delta_k + \alpha \delta_{\bar{k}}}{N - 1 + \alpha} \right)
\]

(3.1)

where \(\xi()\) and \(\mu()\) represent the similarity between the current observation and the reference models,

In the next section, I show the approximation of both conditional probabilities \(\xi(\cdot \mid \cdot)\) and \(\mu(\cdot \mid \cdot)\) based on a common non-parametric statistical test - \(\chi^2\) test. It leads to the improved approach for matching two \(DP-FACT\) features, introduced in the next section.

### 3.2 Matching of DP-FACT

Most existing methods are off-line inference, mainly because the inference is time consuming, for example MCMC (Monte Carlo Markov Chain) sampling method [102] is considered as the standard approach [47]. In order to solve the inference problem in real-time in an on-line manner, the inference of the conditional probabilities may be approximated directly. When it is possible, it relieves the need to calculate the joint probability. Recall the equation of the posterior of the place labelling indicated in equation 3.1. It includes three parts. The last part is a representation of a prior CRP (Chinese Restaurant Process) based on the previous observed labels. It can be calculate directly from the history of measurements. The first two parts are similar. Typically they are approximated by sampling methods. A closer look at them will reveal that they calculate the gamma function of the count of a certain observation over all the possibilities. In other words, they represent the probability of a certain histogram showing up in a sequence of observations. Therefore, it is able to model such a measurement as a measure of the similarity of the current observation to all the pre-defined models. As a result, no sampling methods are needed to estimate this measure if the underlying similarity can be approximated between the current observation and the reference models. This is the basic idea of the online inference method.

#### 3.2.1 DF-FACT Formulation

I name an extended model of \(FACT\) as \(DP-FACT\) [32], which grants \(FACT\) descriptor with statistical meanings. \(DP-FACT\) uses two multinomial distri-
butions, i.e. $DP-FACT_t := \{w_t, g_t\}$ to show the statistical distributions of $Tags$ over discrete feature spaces. $w_t$ is a distribution over the geometrical space (factored by the width of $Tags$), while $g_t$ is over the discretized UV color space. Examples of $g_t$ that extracted from two different nodes are shown in Figure 3.2.1, placed in row order. Intuitively, the distribution in the same row are similar, namely that the difference between rows is greater than that within the same row. The quantitative representation of the differences is given in Section 3.2.

3.2.2 Non-parametric test

Since both observation and existing models are inherently histograms. Therefore the similarity between them can be estimated by non-parametric statistical methods. Here I introduce the approximation of equation 3.1 using $\chi^2$ test.

$\chi^2$ test is formalized as follows [103].

$$\chi^2(m, n) = \sum_{t=1}^{r} \frac{(n_t - N\hat{p}_t)^2}{N\hat{p}_t} + \sum_{t=1}^{r} \frac{(m_t - M\hat{p}_t)^2}{M\hat{p}_t}$$ \hspace{1cm} (3.2)

where $\hat{p}_t = \frac{n_t + m_t}{N + M}$, $N = \sum_{t=1}^{r} n_t$, $M = \sum_{t=1}^{r} m_t$, $r$ is the dimension of both
histograms; \( n_t \) and \( m_t \) are the number of hits at the bin \( t \). The converging condition is \( \sum_{t=1}^{r} p_t = 1 \) according to the definition. For the bins where both histograms have 0 hits, the calculation is skipped.

Deriving from equation (1.8), the estimator of the target label can be approximated as:

\[
p(\phi_t = k | \phi \setminus t, g w) \equiv p(\phi_t | \phi \setminus t) \cdot \xi(w_t | \omega_\phi t) \cdot \mu(g_t | \beta_\phi t) \propto \left( \frac{\sum_{t=1}^{N-1} \delta_{\phi_t} + \alpha \delta_{\phi_k}}{N - 1 + \alpha} \right) e^{-\rho \chi^2(w_t, \omega_k) - (1 - \rho) \chi^2(g_t, \theta_k)}
\]

(3.3)

where \( \rho \in [0, 1] \). For extreme cases, if \( \rho = 1 \), the estimator \( 3.3 \) only considers the geometry measure; for \( \rho = 0 \), only color information is taken. As a reminder of equation 3.1, the two targeting conditional probabilities are formalized as follows.

\[
\begin{align*}
\xi(w_t | \omega_\phi t) & \propto e^{-\rho \chi^2(w_t, \omega_\phi t)} \\
\mu(g_t | \beta_\phi t) & \propto e^{-(1 - \rho) \chi^2(g_t, \theta_\phi t)}
\end{align*}
\]

(3.4)

3.2.3 Model update

Despite the fast calculation, the non-parametric statistic that we introduced in equation 3.3 has an inherent disadvantage. We could see that the non-parametric test is a point estimation without considering history informations. In order to remit this disadvantage, a model update algorithm is used to compensate. Comparing with equation 3.1 where the history information is represented by the counts of occurrences \( n_p^k \) and \( c_q^k \), we require a method to take the history of data into account. It means that the reference model \( \omega_k \) and \( \theta_k \) need to be able to fuse information from all the existing measurements. Instead of saving all the previous observations, I propose an iterative method to fuse the current measurements with existing models as follows.

\[
\begin{align*}
\theta_{k}^{t+1} &= \frac{n_{k}^{t}}{n_{k}^{t} + 1} \theta_{k}^{t} + \frac{1}{n_{k}^{t} + 1} g_t \\
\omega_{k}^{t+1} &= \frac{n_{k}^{t}}{n_{k}^{t} + 1} \omega_{k}^{t} + \frac{1}{n_{k}^{t} + 1} w_t
\end{align*}
\]

(3.5)

where \( n_{k}^{t} \) is the number of frames that have been clustered as with label \( k \) by time \( t \). Therefore, the update process in equation 3.5 is a weighted mean
over the old knowledge and the new observation at each time step. The advantage of this model update algorithm is obvious: In one hand, it can be calculated on-line with low requirements on computational and space costs; on the other other hand, it reflects the history of data in the updated model directly.

3.2.4 Pair-wise Distance for DP-FACT

Following the discussion in Section 2.3 we can analyze the distance matrices of the $\chi^2$ test results and the compound posterior. The result of distance matrix in color space is depicted in Figure 3.2.2(a), and that in width space is shown in Figure 3.2.2(b). An intuitive observation is that color features are able to partition the whole sequence into more segments comparing with width information. $\chi^2$ test leads to more distinctive separations than the results of Euclidean distance. Besides, it is interesting to see that the $\chi^2$ test result for histograms of Tag-width can also indicate the similarity between adjacent frames. It means that the fusion of these information could determine the recognition results more reliably, by introducing multiple constraints.

As previously discussed, tuning the parameter $\rho$ leads to changes in the posterior. It can be observed in Figure 3.2.3 that a higher $\rho$ value, which increases the significance of color features, leads to more possible change-points, since the color features are more salient than width features.

By introducing a median filter, the result is shown in Figure 3.2.4. Note that the color-map of the figure is changed intentionally, for better visibility. The off-diagonal light blobs show that the corresponding frames are similar in appearance. However, this pair-wise result does not imply the number of topological nodes, since the node models keep evolving whenever a new positive reading is detected. Here I just use these plots to reveal the distinctiveness and feasibility of the proposed DP-FACT features. The result of a real-time experiment including all the components of equation 3.1 is shown in the next section.
Figure 3.2.2: Pair-wise distance matrix in UV-color and width space
Figure 3.2.3: Distance matrix using combined features for different $\rho$ values.
Figure 3.2.4: Resulting distance matrix using a median filtering
In this chapter, I introduce experiment results in an indoor office environment. Our approach is compared with keypoint-based methods in terms of labeling accuracy, performance and inference complexity. Two samples of the unwrapped sample images are shown in Figure 5.

Figure 5: Sample images for the omnidirectional camera dataset

4.1 Comparison in Accuracy

As described in [83], the SIFT feature demonstrates a superior accuracy in scene transition detection and recognition accuracy than CENTRIST and the Texture-based method. In this section, we compare the proposed DP-FACT with SIFT as well as a newly developed lightweight keypoint descriptor BRISK [104]. These two descriptors represent the most sophisticated and novel state-of-the-art binary descriptors respectively. It has been reported [104] that the BRISK feature is around 15 times faster than SURF [30] features considering feature extraction and matching, at the same time with similar or even better performances.
As for the keypoint-based methods (SIFT and BRISK), I use the unwrapped images as inputs. The algorithm is designed as follows. Firstly, keypoint-based feature extraction is performed on the input images; then we match the current image with reference images which have been observed in the past and try to get the most similar reference; if the ratio of the number of positively matched features and the number of features extracted from reference images is above a given threshold (in this case 70%), I label the current image with the same label as the best matched reference; otherwise I consider that the current image has a new label by taking it as a newly added reference image.

The test result is shown in Figure 4.1.1. In order to ease the comparison, the figures are aligned in time series. The first two plots on the top are the raw output of DP-FACT and the result after applying an online median filter over the past 5 frames. Please notice that further off-line smoothing of the labeling can be implemented as well [105, 106], which potentially provides more precise results. The result of keypoint-based methods after the same median filtering are given in the third plot. Specifically, the vertical axis of the top three subfigures in Figure 4.1.1 indicates the online recognized labels.

The results indicated by the first three plots in Figure 4.1.1 show that the proposed DP-FACT framework leads to more stable outputs than keypoint-based methods. On the contrary, keypoint-based methods have high false positive ratio on the transition detection, because the labeling is dominated by massive changes of keypoints even in the same scene. As a result, high scene change rate was observed and the number of scenes detected from the sequence is much higher. Moreover, using the proposed method, the detection of new places is automatic, except setting the weighting factor for different information.

The “Compressed Image Sequence” stripe shows a squeezed summary of the whole image sequence, from which the scene change can be intuitively observed. The “Experiment Result” line shows the filtered output of DP-FACT. The “Transition areas” indicate that the robot is closely passing a doorway or turning corners, where the scene recognition does not make much sense and therefore not considered in the statistical results. The corresponding behavior of the algorithm is that the output label can hardly be stable even after median filtering, which can be readily detected and labeled.

The “Ground Truth” is manually labeled by only observing the input video sequences. It means that the images with the similar appearance are considered to be obtained from the same scene. Compared with “Experiment Result”, we could infer that the change point detection is more practical than the keypoint-based approaches. An overlaid 2D sketch of the experiment
Figure 4.1.1: Experiment results. From top to bottom: raw label output of DP-FACT; result of DP-FACT after median voting filter of 5 frames; result of keypoint-based approaches (SIFT, BRISK) after median filtering; image sequence in a compressed layout; labeled ground truth; result of DP-FACT; label explanations; an overlaid sketch of the test environment by detected scene appearances.
environment by the experiment result of $DP$-$FACT$ is depicted in the bottom of Figure 4.1.1. The image sequence starts from the right side of the map. Sample images from different scenes are illustrated around the sketch, showing the appearance differences of various scenes. Although with some mis-classifications, $DP$-$FACT$ shows more reliable and feasible results for the scene recognition process.

<table>
<thead>
<tr>
<th>Method</th>
<th>Recognition Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIFT</td>
<td>73.3%</td>
</tr>
<tr>
<td>BRISK</td>
<td>66.7%</td>
</tr>
<tr>
<td>$DP$-$FACT$</td>
<td>89.4%</td>
</tr>
</tbody>
</table>

Table 4.1.1: Accuracy of scene recognition

As part of qualitative comparison, table 4.1.1 shows the accuracy of scene recognition. Because the transition detection for keypoint-based method is vague, leading to frequent false positives, the scene recognition results for keypoint-based methods are calculated by considering non-repeated labels in the same scene as a group. Since $FACT$ requires an off-line filtering, the comparison is not included. We can see that $DP$-$FACT$ has the best recognition accuracy, though color is a relatively “weaker” feature than keypoint descriptors.

Two possible reasons why keypoint-based methods perform worse can be considered: First, the distortion of the uncalibrated omnidirectional images causes non-uniform resolution of the unwrapped images, which makes the keypoint-based feature extraction unstable, especially when the keypoints are at different distances. Second, $DP$-$FACT$ is structured only in horizontal direction, where the information is summarized in one dimension. However, keypoints can be possibly detected on the whole 2D surface of the image. This consistency of feature construction maximizes the difference between any two labels and more importantly minimises the influence of unexpected randomness.

4.2 Evaluation in time cost

The evaluation of time cost is shown in Figure 4.2.1. Because the number of nodes rises during the test, we see that the overall time slightly rises as well. Comparing with the time cost of common sampling methods, the gray area
in Figure 4.2.1 indicates that the inference time of the proposed estimation is less than 5 millisecond.

![Figure 4.2.1: Time cost of DP-FACT over frames. The lines are filtered results out of raw measurements (in circles). The gray area indicates the inference time.](image)

Here I show a further study of the relation between the inference time and the complexity of the model. Figure 4.2.2 depicts a regression result of the inference time over the number of nodes, which is substantially linear. This result implies the potential of the proposed method can be extended to a large scale environment without jeopardizing the realtime ability.

Recall the test in Figure 4.1.1. In addition to the superior recognition accuracy, DP-FACT shows faster performance. Figure 4.2.3 depicts the comparison in time.

Our aim is to develop an online scene recognition algorithm which can be implemented online with limited computational resources. I evaluated the algorithm on three different types of CPUs in order to show that the method is feasible for different applications. The result is shown as Figure 4.2.4. We see that even for low speed CPUs, the algorithm can still reach around 17Hz.

### 4.3 Discussion on $\chi^2$ test and naive matching

Here I discuss the theoretical reason why $\chi^2$ test provides more distinct results, e.g. by comparing Figure 5 and Figure 7(a). Considering a summation
Inference time in ms

where $\mu$ be interpreted that the covariance of the features is not considered.

form of Euclidean distances, for two feature vectors $x_1$ and $x_2$, the Euclidean distance is represented by:

$$d_e = \sqrt{(x_1 - x_2)^T(x_1 - x_2)} = \sqrt{(x_1 - x_2)^TI_{kk}(x_1 - x_2)}$$

(4.1)

where $I_{kk}$ is an identity matrix with the same dimension as the feature. It can be interpreted that the covariance of the features is not considered.

On the other hand, considering the $\chi^2$ test introduced in equation 3.2, it has a limiting $\chi^2$ distribution with $N + M - 1$ degrees of freedom. Under null hypothesis it has mean vector $\mu$ and covariance matrix $V$ as follows:

$$\mu = (N + M) \times (\hat{p}_1, \hat{p}_2, \ldots \hat{p}_r)^T$$

(4.2)

$$V = (N + M) \begin{pmatrix}
\hat{p}_1(1 - p_1) & -\hat{p}_1\hat{p}_2 & \ldots & -\hat{p}_1\hat{p}_r \\
-\hat{p}_2\hat{p}_1 & \hat{p}_1(1 - \hat{p}_1) & \ldots & -\hat{p}_2\hat{p}_r \\
\vdots & \vdots & \ddots & \vdots \\
-\hat{p}_r\hat{p}_1 & -\hat{p}_k\hat{p}_2 & \ldots & \hat{p}_r(1 - \hat{p}_r)
\end{pmatrix}$$

(4.3)
using the notation of equation 3.2. This additional information enables that all the Tag’s in a FACT descriptor are jointly considered, respecting with the frequency of hits on each discrete bin of the feature space.

4.4 Further experiments on public dataset

In order to further validate the results, I apply the proposed DP-FACT onto a widely cited dataset called COLD [1]. It is a collection of indoor omni-directional images from Freiburg, Ljubljana and Saarbrücken. Since the white-balance for those images could not be properly adjusted in an online manner, I use only the ones captured in cloudy weather in order to minimize the influence on standard color. Statistical results are illustrated in table 4.4.1. The definition of each column is explained as follows:

- **Dataset**: name of the test dataset;
- **#Image**: #images included in the dataset;
- **#SemNode**: #provided labeled semantic nodes;
- **#Node**: #detected nodes using DP-FACT;
- **#Tran**: detected #scene transitions;
- **#Tag**: average #detected Tags per image;
- **Total Time**: statistics of the total time cost per image;
- **Inference Time**: statistics of the inference time per image;
Figure 4.2.4: Performance on different CPUs, using single core
<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Image</th>
<th>#SemNode</th>
<th>#Node</th>
<th>#Tran</th>
<th>Total Time (ms)</th>
<th>Inference Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Mean</td>
<td>Max</td>
</tr>
<tr>
<td>Freiburg PathA</td>
<td>1459</td>
<td>5</td>
<td>10</td>
<td>17</td>
<td>25.3</td>
<td>15.03</td>
</tr>
<tr>
<td>Ljubljana PathA</td>
<td>1871</td>
<td>6</td>
<td>9</td>
<td>22</td>
<td>19.1</td>
<td>13.34</td>
</tr>
<tr>
<td>Saarbrücken PathA</td>
<td>2941</td>
<td>8</td>
<td>14</td>
<td>19</td>
<td>20.5</td>
<td>14.39</td>
</tr>
<tr>
<td>Saarbrücken PathB</td>
<td>1306</td>
<td>5</td>
<td>9</td>
<td>11</td>
<td>21.6</td>
<td>14.64</td>
</tr>
</tbody>
</table>

Table 4.4.1: Test result using COLD datasets

Table 4.4.1 shows that DP-FACT can lead to reasonable segmentation of the environment with low computational time. The Freiburg dataset leads to higher total time cost. It is because there are more vertical lines detected, and the additional time cost is due to descriptor construction. Note that the number of detected nodes (i.e. #Nodes) is usually higher than the number of manually labeled semantic nodes, because, for most cases, the manually labeled semantic areas, such as “a corridor”, may contain multiple appearances. Taking Figure 4.4.1 as an example, where the standard trajectory for Path A is used [1], we see that the corridor labeled by ② is actually segmented into five regions with respect to door positions etc. Different colors indicate the segmentation results. It demonstrates that the proposed appearance-based DP-FACT can lead to precise representation of a target environment.
Figure 4.4.1: Segmentation result based on DP-FACT for Freiburg Path A [1].
I have presented a framework to solve SCAR problem in near-realtime with synchronized multi information. Based on that, a real-time scene recognition method was developed, which depends on a lightweight color based descriptor, achieving topological modeling of indoor environments using omnidirectional cameras. I proposed to use a Dirichlet Process Mixture Model (DPMM) to manage new scene registration and recognition simultaneously. The experiment results show the advantage of the proposed framework in the sense of online computation ability and better recognition performance than keypoint-based methods. This study also shows that the inference of a DPMM can be approximated by reasoning the conditional probabilities directly. We envision that similar concepts can adopted to solve other inference problem with large target space as well. It is also possible to use such models for data modeling problem with multiple observations.

We must notice that the proposed FACT descriptor only deals with indoor environments, where vertical lines are preserved in the field of view of unwrapped panoramic images obtained by omnidirectional cameras. Therefore, the results do not imply that the extended applications for semi-structured environment is easily feasible. Not withstanding this limitation, this work does suggest that color based features can be integrated to a real-time online scene recognition and topological mapping robotics system, with relatively good performance. We can imagine the combination of keypoint- and color-based methods will help to solve this problem at a hybrid level, without limiting the targeting environment. Regarding loop-closing problem, the proposed framework can help the selection of target poses to be matched, with low computational cost. For example, the loop-closing can be only performed to the poses in the scenes which have the same label as the current image frame. The conducted results will be shown in future work.

Besides, we see that the selection of weights $\rho$ is essential for modeling. Therefore an automatic tuning algorithm based on either covariance analysis or non-parametric density estimation can be envisaged for the next steps.
Four different homing algorithms are compared.

Experiments are carried out in several indoor environments: apartments, offices, doorways etc.

Real-time online implementation of navigation system based on finite state-machine.

Comparison of different descriptors, such as SIFT, BRISK etc.

Unless we change our direction, we will end up where we are heading.

*Chinese Proverb*
Chapter 6
Visual homing and visual servoing

SCENE recognition algorithm using DPMM has been discussed in the previous part. The outcome resulted in a topological segmentation of an environment. Based on the topological representation, the navigation ability of the robot is generally required in order to fulfill common services. Topological navigation has been widely applied for such scenarios. In this part, I demonstrate a generic framework for visual homing methods by adopting visual servoing concepts. Besides, an integrated framework for visual topological navigation is developed using an omnidirectional camera.

6.1 Visual Servoing Framework

6.1.1 Principles of Visual Servoing

In this chapter, I assume that the robot can be controlled by a velocity vector, including directions and absolute values of the speed. This neglects the non-holonomic properties of most robotic platforms. However, it is acceptable for the simple differential robots used in experiments. More work would be needed to adapt this control framework to a more complex system such as a car or a space rover.

When we consider the homing problem as a control problem in the appearance space, it can be summarized as an IBVS problem. In this context, the objective is to drive an error $e$ between those derived from observed and desired appearances to zero. In the classical IBVS, the error would be the difference in feature coordinates (in pixel). According to the fundamentals of visual servoing, this error can be minimized, given that the error dynamics is linked to the control input $v$ using an interaction matrix $L_e$ and the following relation [107],

$$\dot{e} = L_e v \quad (6.1)$$

Once we have calculated the error, a direct P-controller can be set up for the motion control of the robot. The controller is designed to eliminate or
minimize the error $e$ with

$$v = -\lambda L_e^+ e$$

(6.2)

where $L_e^+$ is the pseudo-inverse of the interaction matrix $L_e$. This controller is designed to have an exponential convergence rate of the error, if the stability of the system is ensured. According to the stability analysis of IBVS in [107], the interaction matrix and its pseudo inverse need to be full rank, in order to guarantee local asymptotic stability.

### 6.1.2 Visual Homing

Visual homing is often implemented using bearing-only methods. An initial work was presented by Cartwright and Colletti [108] as the ‘snapshot’ model. Franz et al. [109] continued this direction by analyzing the error and convergence properties. In my previous work [20], I gave a proof of the convergence of a simplified bearing-only method, based on Lyapunov stability theory. In this chapter, I formulate the bearing-only problem [20] in the classical IBVS framework and I propose three other different homing methods under this framework.

This method is stimulated by the work of Corke et al. [110], where the authors used the ALV [111] (Average Landmark Vector) principle to realize visual servoing tasks. The ALV-based method converts the homing problem to vector operations, by summing up the bearing vectors to a number of key-points at the reference and current position. The difference between these two sums is then used to compute the homing vector. Two limitations should be considered: firstly, outliers will greatly affect the accuracy of the homing vector, and secondly, this approach depends on knowing the position of the landmarks used as key-points. This in turns assumes that it is possible to estimate the distance to the key-points. In comparison, this approach takes advantage of the scale information attached to the key-points to calculate the homing vector without distance estimation and with a computational requirement as low as possible.

According to Goedeme et al. [19], knowing the structure of the environment and in particular the landmark positions is not necessary for visual homing. This information can be recovered by estimating the ratio of the distances to the matching key-points by triangulation using an Extend Kalman Filter, if needed. Using the features’ scales, we can avoid this estimation step and use the scale variation as a proxy for the distance error.

Recently, Lim et al. [112] presented a homing algorithm based on the following principle: they divide the 2D plane into four regions and estimate the current robot position by measuring the bearings to known landmarks.
Compared with their approach, I prove the convergence of the method using Lyapunov theory. It guarantees the stability of the controller in a general mathematical sense. Moreover, the observability analysis is also provided, which shows that the robot displacement is observable using the scale information under known velocity commands.

One of the latest work in visual homing [113] was proposed by Aranda et al, which improved the performances by using 1D trifocal tensors from the omnidirectional camera. Compared with the works using 1D-trifocal tensors [114, 115], which relies on three view geometry, the method infers the homing vector directly from the current appearance, and as result is less reliant on feature association. Besides, the approach does not require solving non-linear equations constructed from the tensors. If the reconstruction of the environmental structure is not needed, the proposed servoing-based method requires less computational power.

Furthermore, [116] used a sliding-mode control law to exploit the epipolar geometry; [117] directly calculated the homographies from raw images; Cherubini et al. [118] proposed a redundant framework for visual homing problem, which, in particular, allows online obstacle avoidance. The comparison with these works is not considered in this thesis, since the basic strategies and premises are significantly different, e.g. all these methods require a global representation of the environment.

Some related early work using SIFT as main features for visual homing was proposed in [119, 120]. They considered the epipolar geometries as well as the orientation and the scales of SIFT features for monocular cameras, using a framework similar to [39]. Among those, the work by Vardy et al. [121] is the closest to the simplified approach using scale information. Their first work developed a scale invariant local image descriptor for visual homing, based on the optical flow of unwrapped panoramic image from an omnidirectional camera. This work was continued by Churchill et al. [122], which presented results of real-time homing experiment using the scale difference field in panoramic images, computed from SIFT matches. In comparison to their work, I stress the following two main differences: firstly, I carry out reasoning on the effect of the change of scales in a more dedicated way, by embedding the scale measures inside the visual servoing framework. Secondly, I give a mathematical proof of the convergence of the controller. We will refer to their method as Heuristic Scale-based Visual Servoing (HSVS) in the following and show the proposed approach outperforms theirs in terms of precision.
6.1.3 Integration and Control Loop

Usually the visual homing algorithms can be instantiated on a real system as depicted in Figure 6.1.1. The visual control processing pipeline is depicted on right hand side of the dotted line. It starts with capturing a new panoramic image. During each running cycle, the omnidirectional image acquired from the camera is unwrapped first, using a simple polar-coordinate transformation. Note that this implies a minor assumption on the calibration of the camera, namely the image center is known, but it is not necessary to know a full model of the catadioptric shape.

SIFT features are then extracted from the unwrapped images and used to compare the current image with the one acquired at the home position. Although there is certainly distortion in the SIFT features due to mirror non-linearity, this approach is also used in related works and in particular in [121].

From the matched key-points, the visual servoing control law is applied, and a homing vector is computed. This vector is then transmitted to the robot.
motion controller. The control vector is then implemented while accounting for the non-holonomic constraints of the target robot. The control loop is executed with each newly captured image until convergence.

6.1.4 Visual homing by omnidirectional cameras

The visual homing problem can be defined as shown in Figure 6.1.2, where \( p_1, p_2, \ldots, p_n \) are \( n \) key-points, which can be extracted by SIFT, SURF [30] or other methods that provide scale information of key-points. It is assumed that enough number of key-points (three points for the 2D case) can be seen from the current position \( C \) and the home position \( O \). The objective is to guide the robot from \( C \) to \( O \) by only knowing each observed scale \( s_i \) and bearing angle \( \beta_i \) associated with each key-point \( p_i \). Negre et al. [123] showed that the intrinsic scale can be a measurement of the time to collision. Authors of [120] showed a direct relation between the scale and the distance to the feature point. However, for different setups and different environments, the absolute distances to the features cannot be mapped directly. We believe that the variation of the scale of a keypoint can be seen, in first approximation, as a proxy for the variation of its distance.

One fundamental reason is that the scale of a key-point, in number of pixels, corresponds to real physical size of a patch imaged from the camera. Using a standard pin-hole model of the camera, the scale is actually inversely proportional to the distance between the camera and the imaged patch, assuming the object scale and focal length are constants. As a result, the error between the observed scale and the reference scale gives us an indication of the error between the respective distances and can be used to control the robot towards the home position.

6.2 Image Based Visual Servoing using uncalibrated omnidirectional cameras

In this chapter, it is intended to use the visual servoing framework to build a robot controller, by using the scale and bearing to the key-points, instead of their coordinates. If the key-points are extracted from an omnidirectional camera, we can easily convert image coordinates to bearing angles. Meanwhile, the robot configuration can be summarized by its position \((x, y)\) and its heading \(\theta\).
Figure 6.1.2: Abstracted problem of homing. Keypoints $p_1$ to $p_n$ can be observed at the current position $C$ and at the Home position $O$. The variant sizes indicate the differences of key-points in scale.

6.2.1 Definitions of Visual Error

The error of the system is made of two components: the scale errors and the bearing angle errors. Therefore the vector of the error can be written as:

$$e = (s - s^*, \beta - \beta^*)^T$$

(6.3)

where $s = (s_1, \ldots, s_n)$ is the vector of observed scale of the key-points and $\beta = (\beta_1, \ldots, \beta_n)$ is the vector of their bearing angles. The variables with star-superscripts are reference values.

Before computing the derivative of the error, we need to derive the relation between the scale of a feature $s_i$ and the distance to the corresponding object $l_i$. Let us use $f$ to denote the focal length of the camera\(^1\) and $S$ the physical size of the region defined by the corresponding key-point patch. Using simple triangulation and the camera pin-hole model, we have

$$s_i = \frac{S f}{l_i} \quad \text{and} \quad s^*_i = \frac{S f}{l^*_i}$$

(6.4)

\(^1\)In the case of an unwrapped catadioptric image, I abuse the notation by relating the focal length to the vertical field of view $\alpha$ and the image height $h$: $\frac{\alpha}{2} = \text{atan2}(h, f)$.
which leads to
\[ s_i = s_i^* \frac{l_i^*}{l_i} \quad (6.5) \]

If we assume that the physical key-point \( i \) is at the 2D coordinates \((x_i, y_i)\) in the same frame as the robot coordinates, we can also explicit the relation between \((l_i, \beta_i)\) and the robot coordinate \([x, y]^T\):
\[
l_i = \sqrt{(x_i - x)^2 + (y_i - y)^2} \quad (6.6)
\]
\[
\beta_i = \text{atan2}(y_i - y, x_i - x) - \theta \quad (6.7)
\]

### 6.2.2 Error derivative

To compute the error derivative \( \dot{e} \), I compute independently the scale and bearing derivatives, by considering them as functions of the robot pose.

\[
\frac{d}{dt} [s_i(x, y, \theta) - s_i^*] = \frac{d s_i}{dx} \frac{dx}{dt} + \frac{d s_i}{dy} \frac{dy}{dt} + \frac{d s_i}{d\theta} \frac{d\theta}{dt} \quad (6.8)
\]

Using equations 6.5 and 6.6, we have
\[
\frac{d}{dt} [s_i(x, y, \theta) - s_i^*] = s_i^* l_i^* \left[ v_x \frac{d}{dx} \frac{1}{l_i} + v_y \frac{d}{dy} \frac{1}{l_i} \right]
\]

\[
= -s_i^* l_i^* \frac{1}{l_i^2} \left[ v_x \cos \beta_i^g + v_y \sin \beta_i^g \right] \quad (6.9)
\]

with \( v_x = \frac{dx}{dt} \) and \( v_y = \frac{dy}{dt} \).

Similarly, the bearing error can be derived as
\[
\frac{d}{dt} [\beta_i^g(x, y, \theta) - \beta_i^*] = \frac{d \beta_i^g}{dx} \frac{dx}{dt} + \frac{d \beta_i^g}{dy} \frac{dy}{dt} + \frac{d \beta_i^g}{d\theta} \frac{d\theta}{dt} \quad (6.10)
\]

Using equations 6.7 this leads to
\[
\frac{d}{dt} [\beta_i^g(x, y, \theta) - \beta_i^*] = -\frac{y_i - y}{l_i^2} v_x - \frac{x_i - x}{l_i^2} v_y - \omega
\]
\[
= -\frac{1}{l_i} \left[ v_x \sin \beta_i^g + v_y \cos \beta_i^g \right] - \omega \quad (6.11)
\]

with \( \omega = \frac{d\theta}{dt} \).

Similarly, the derivative of the distance to a key-point \( i \) is,
\[
\frac{d}{dt} [l_i(x, y, \theta) - l_i^*] = -(v_x \cos \beta_i^g + v_y \sin \beta_i^g) \quad (6.12)
\]
Note that $\beta_i^g$'s are bearing angle of feature point $i$ in the global frame instead of robot local frames. In order to transform them to the robot local frame, the heading difference needs to be considered, such that the real bearing observations by the robot are:

$$\beta_i = \beta_i^g + (\theta - \theta^*)$$  \hspace{1cm} (6.13)

where we consider $\theta^* = 0$.

Combining equations 6.9 and 6.11, we can write the error dynamics as follows:

$$\frac{d}{dt} e = L_e v$$  \hspace{1cm} (6.14)

\[
\begin{pmatrix}
    s_1 - s_1^* \\
    \vdots \\
    s_n - s_n^* \\
    \beta_1 - \beta_1^* \\
    \vdots \\
    \beta_n - \beta_n^*
\end{pmatrix} = \begin{pmatrix}
    \frac{s_1^* l_1}{l_1^*} \cos \beta_1 & \frac{s_1^* l_1}{l_1^*} \sin \beta_1 & 0 \\
    \vdots & \vdots & \vdots \\
    \frac{s_n^* l_n}{l_n^*} \cos \beta_n & \frac{s_n^* l_n}{l_n^*} \sin \beta_n & 0 \\
    -\frac{1}{l_1^*} \sin \beta_1 & -\frac{1}{l_1^*} \cos \beta_1 & -1 \\
    \vdots & \vdots & \vdots \\
    -\frac{1}{l_n^*} \sin \beta_n & -\frac{1}{l_n^*} \cos \beta_n & -1
\end{pmatrix} \begin{pmatrix}
    v_x \\
    v_y \\
    \omega
\end{pmatrix}
\]

As mentioned earlier, the interaction matrix in equation 6.14 is sufficient to implement a visual servoing controller. One remaining problem is that neither the distances $l_i$ nor $l_i^*$ can be quantified easily using a single camera. Based on [124] and the analysis on the errors in [40], these values can be approximated by constants due to the low sensitivity of the controller to these parameters. The validity of this assumption will be shown in simulation.

A direct way to reduce the complexity is by noticing that either the upper part or the lower part of equation 6.14 are sufficient to complete a visual servoing task. As it is trivial to rotate the robot on the spot once the translational error has been corrected, a two-stage controller can be considered. I consider only the first stage here - the translation to the home position, because it is the key issue for homing. In practice, this means that we can either implement a scale-only visual servoing or a bearing-only visual servoing. Accordingly, the interaction matrix for scale-only visual servoing is shown as follows:

$$\frac{d}{dt} \begin{pmatrix}
    s_1 - s_1^* \\
    \vdots \\
    s_n - s_n^*
\end{pmatrix} = \begin{pmatrix}
    \alpha_1 s_1^* \cos \beta_1 & \alpha_1 s_1^* \sin \beta_1 \\
    \vdots & \vdots \\
    \alpha_n s_n^* \cos \beta_n & \alpha_n s_n^* \sin \beta_n
\end{pmatrix} \begin{pmatrix}
    v_x \\
    v_y
\end{pmatrix}$$  \hspace{1cm} (6.15)
Controller based on this interaction matrix will be denoted as \text{SOVS} in the remainder of this chapter. A similar method, which uses the lower part of equation 6.14, is a bearing-only approach denoted by \text{BOVS}. The error dynamics can be derived as:

\[
\frac{d}{dt} \begin{pmatrix}
\beta_1 - \beta_1^* \\
\vdots \\
\beta_n - \beta_n^*
\end{pmatrix} =
\begin{pmatrix}
\gamma_1 \sin \beta_1 & \gamma_1 \cos \beta_1 \\
\vdots & \vdots \\
\gamma_n \sin \beta_n & \gamma_n \cos \beta_n
\end{pmatrix}
\begin{pmatrix}
v_x \\
v_y
\end{pmatrix}
\]  (6.16)

According to the generic properties of the controller stated in Section 6.1.1, the local asymptotic stability will be maintained if each interaction matrix and its pseudo inverse are full-ranked. This can be ensured using a reasonable large number of matched features in real applications.

Recalling equation 6.13, the estimation of the heading \(\theta\) is crucial for the calculation of the interaction matrices. It implies that a robot may need to have absolute references such as a magnetic compass or a reliable visual compass for better accuracy. Regarding the dataset in Section 9.2, where the robot is well aligned, this problem is trivial. However, this matter needs to be considered in real applications, such as using a visual compass.

6.3 Observability Analysis

6.3.1 Definitions

Given the raw sensor measurements, the observability analysis of system states is important before we step on the designation of other advanced controllers. The observability analysis also gives hints for what results can be expected from the system configuration. The configuration of the discussed IBVS system is described via the following system states, which are close to the target variables for observability analysis of the feature distributions and
In order to get the full state for a motion constrained in 2D plane, we require at least three positively matched key-points, considering the number of geometrical constraints. Without loss of generality, we can use the following subset of the full state, where the minimum required keypoints are denoted by subscripts 1 2 and 3.

\[
x = \begin{bmatrix}
x : x\text{-coordinate of the robot position} \\
y : y\text{-coordinate of the robot position} \\
\theta : \text{heading of the robot} \\
s : \text{vector of observed scales} \\
\beta : \text{vector of observed bearings} \\
l : \text{distance to features}
\end{bmatrix}
\] (6.17)

Following the definitions in equations (6.9) (6.11) and (6.12), using star-marks to denote the reference variables, the state derivatives of the system state is

\[
x = (x, y, \theta, s_1, s_2, s_3, \beta_1, \beta_2, \beta_3, l_1, l_2, l_3)^T
\] (6.18)
presented by equation (6.19).

\[
\dot{x} = \begin{pmatrix}
\cos \theta & \sin \theta & 0 \\
-\sin \theta & -\cos \theta & 0 \\
0 & 0 & 1 \\
-s_1^* l_1^* \cos (\beta_1 + (\theta - \theta^*)) & -s_1^* l_1^* \sin (\beta_1 + (\theta - \theta^*)) & 0 \\
-s_2^* l_2^* \cos (\beta_2 + (\theta - \theta^*)) & -s_2^* l_2^* \sin (\beta_2 + (\theta - \theta^*)) & 0 \\
-s_3^* l_3^* \cos (\beta_3 + (\theta - \theta^*)) & -s_3^* l_3^* \sin (\beta_3 + (\theta - \theta^*)) & 0 \\
-\sin (\beta_1 + (\theta - \theta^*)) l_1 & -\cos (\beta_1 + (\theta - \theta^*)) l_1 & -1 \\
-\sin (\beta_2 + (\theta - \theta^*)) l_2 & -\cos (\beta_2 + (\theta - \theta^*)) l_2 & -1 \\
-\sin (\beta_3 + (\theta - \theta^*)) l_3 & -\cos (\beta_3 + (\theta - \theta^*)) l_3 & -1 \\
\end{pmatrix}
\begin{pmatrix}
v_x \\ v_y \\ \omega
\end{pmatrix}
\] (6.19)

The observation function, which is also the 0-order Lie derivative, includes the scale and bearing information:

\[
L^0 h = h(x) = \begin{pmatrix}
s_1 & s_2 & s_3 & \beta_1 & \beta_2 & \beta_3
\end{pmatrix}^T
\] (6.20)

The further Lie derivatives of the observation function over the control functions are shown in (6.21) - (6.25). The control functions are comprised of the three columns of (6.19), denoted by \(f_1, f_2, f_3\) respectively. For the purpose of conciseness, I use \(C_i\) to denote \(\cos (\beta_i + \theta - \theta^*)\), and \(S_i\) for \(\sin (-\beta_i - \theta + \theta^*)\).

\[
\nabla L^0 h = \nabla h(x) = \begin{pmatrix}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\] (6.21)
6.3.2 Observation matrix

We stop the computation by (6.25), because they have illustrated that the observation matrix shown in equation (6.22) is already row full ranked using equations (6.21) (6.24) and (6.25), but not column full ranked. We notice that continuing calculating further derivatives would not help it become column full rank, since the entries are independent of $x$ or $y$.

\[
M = \begin{pmatrix}
\nabla L^0 h \\
\nabla L^1_{f_1} h \\
\nabla L^1_{f_2} h
\end{pmatrix}
\]  
(6.22)

Using all the measurable variables and the known control commands, the analysis of the null space of $M$ reveals the inherent relations among each state. The null space of the full observation matrix $M$ is given in (6.23).

\[
\text{nullspace}(M) = \{[1, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0]\}
\]  
(6.23)

It shows that the position of the robot $(x, y)$ in the global frame is not observable due to non-zero entries in the nullspace. It is acceptable, since they are not of interest without knowing the global reference frame. More importantly, it shows that the other states $\theta, s, \beta$ and particularly the distances to the feature points $l$ are observable, as long as the control command to the robot is known.

Though not explicitly calculated by the homing algorithm, these observable constraints imply that it is able to determine the translation and rotation to a predefined position, since three known distances to static feature points are adequate to well-determine such a transform for the motion in a 2D plane.
$$L_{f_1}^1 h = \frac{\partial h(x)}{\partial x} \cdot f_1 = \nabla L_{f_1}^0 h \cdot f_1 = \left( -\frac{s_1 l_s^* C_1}{l_1^2}, -\frac{s_2 l_s^* C_2}{l_2^2}, -\frac{s_3 l_s^* C_3}{l_3^2}, \frac{S_1}{l_1}, \frac{S_2}{l_2}, \frac{S_3}{l_3} \right)^T$$

$$\nabla L_{f_1}^1 h = \begin{pmatrix}
0 & 0 & -\frac{s_1 l_s^* S_1}{l_1^2} & 0 & 0 & -\frac{s_2 l_s^* S_1}{l_2^2} & 0 & 0 & 2 \frac{s_1 l_s^* C_1}{l_3^2} & 0 & 0 \\
0 & 0 & -\frac{s_2 l_s^* S_2}{l_2^2} & 0 & 0 & 0 & -\frac{s_2 l_s^* S_2}{l_2^2} & 0 & 0 & 2 \frac{s_2 l_s^* C_2}{l_2^3} & 0 \\
0 & 0 & 0 & 0 & 0 & -\frac{s_3 l_s^* S_3}{l_3^2} & 0 & 0 & 2 \frac{s_2 l_s^* C_3}{l_3^3} & 0 & 0 \\
0 & -\frac{C_1}{l_1} & 0 & 0 & -\frac{C_1}{l_1} & 0 & 0 & -\frac{S_1}{l_1^2} & 0 & 0 \\
0 & -\frac{C_2}{l_2} & 0 & 0 & 0 & \frac{C_2}{l_2} & 0 & 0 & -\frac{S_2}{l_2^2} & 0 \\
0 & -\frac{C_3}{l_3} & 0 & 0 & 0 & 0 & \frac{C_3}{l_3} & 0 & 0 & -\frac{S_3}{l_3^2} \\
\end{pmatrix} \quad (6.24)$$

$$L_{f_2}^1 h = \frac{\partial h(x)}{\partial x} \cdot f_2 = \nabla L_{f_2}^0 h \cdot f_2 = \left( -\frac{s_1 l_s^* S_1}{l_1^2}, -\frac{s_2 l_s^* S_2}{l_2^2}, -\frac{s_3 l_s^* S_3}{l_3^2}, -\frac{C_1}{l_1}, -\frac{C_2}{l_2}, -\frac{C_3}{l_3} \right)^T$$

$$\nabla L_{f_2}^1 h = \begin{pmatrix}
0 & 0 & -\frac{s_1 l_s^* S_1}{l_1^2} & 0 & 0 & -\frac{s_1 l_s^* S_1}{l_1^2} & 0 & 0 & 2 \frac{s_1 l_s^* S_1}{l_1^5} & 0 & 0 \\
0 & 0 & -\frac{s_2 l_s^* S_1}{l_2^2} & 0 & 0 & 0 & -\frac{s_2 l_s^* S_1}{l_2^2} & 0 & 0 & 2 \frac{s_2 l_s^* S_1}{l_2^3} & 0 \\
0 & 0 & 0 & 0 & 0 & -\frac{s_3 l_s^* S_3}{l_3^2} & 0 & 0 & 2 \frac{s_3 l_s^* S_3}{l_3^3} & 0 & 0 \\
0 & -\frac{S_1}{l_1} & 0 & 0 & -\frac{S_1}{l_1} & 0 & 0 & -\frac{C_1}{l_1^2} & 0 & 0 \\
0 & -\frac{S_2}{l_2} & 0 & 0 & 0 & -\frac{S_2}{l_2} & 0 & 0 & -\frac{C_2}{l_2^2} & 0 \\
0 & -\frac{S_3}{l_3} & 0 & 0 & 0 & 0 & -\frac{S_3}{l_3} & 0 & 0 & -\frac{C_3}{l_3^2} \\
\end{pmatrix} \quad (6.25)$$
Chapter 7

Fast Visual Homing: SSVS

Based on the observability analysis, the scale information and bearing angles from local frame and especially their changes are sufficient information for pose control. Based on that, I describe a scale-based visual homing approach that does not require the computation of the pseudo-inverse of an interaction matrix in this chapter. Above all, this approach is independent of the global heading estimation. Since the global heading is usually approximated by visual compass and with inneglectable error [125], this method avoids such potentially extra errors in real applications. I also provide the convergence proof for the resulted control law.

7.1 Scale-based Control for a 1-D Robot

Recalling equation 6.9:

$$\frac{d}{dt} (s_i - s_i^*) = -\frac{s_i^* l_i^*}{l_i^2} [v_x \cos \beta_i + v_y \sin \beta_i]$$

For the sake of argument, let us first consider a 1D mobile robot, which is only able to move along the direction towards a certain key-point indexed by $i$. Because the right side of the above equation can be seen as the projection of the robot velocity in the direction towards the key-point, denoting $e_i = s_i - s_i^*$ and $v_i = v_x \cos \beta_i + v_y \sin \beta_i$, we have:

$$\frac{d}{dt} e_i = -\frac{s_i^* l_i^*}{l_i^2} v_i$$ (7.1)

Following the typical designation strategy of visual servoing, we would like to ensure an exponential decoupled decrease of the error [107]. The following trivial control would achieve this goal (where $\lambda_i$ is a positive constant).

$$v_i = \lambda_i e_i$$ (7.2)
### 7.2 Generic Scale-based Control

When abusing the intuition that several individual controllers for the 1D case may be combined together to calculate the required velocity for the 2D case, we have:

\[
\begin{pmatrix}
    v_x \\
v_y
\end{pmatrix} = \sum_{i=1}^{n} \lambda_i \left( s_i - s_i^* \right) \begin{pmatrix}
    \cos \beta_i \\
    \sin \beta_i
\end{pmatrix} \tag{7.3}
\]

However, even if the convergence was obvious in the 1-D case, there is no guarantee that this sum of control contributions would lead to a stable controller. In order to show the convergence, I resort to the Lyapunov theory. We define the following non-negative energy function (Lyapunov candidate function):

\[
E = \frac{1}{2} \sum_{i=1}^{n} \left( \frac{s_i - s_i^*}{s_i^*} \right)^2 \tag{7.4}
\]

In this autonomous system with \( n \)-dimensional states \( s \), the only equilibrium is where \( s = s^* \) in the feature space; and physically it is the reference home position. According to Lyapunov theory, the following condition needs to be shown:

\[
\begin{cases}
    \frac{d}{dt} E(t) = 0, \text{ only when all } s_i = s_i^* \\
    \frac{d}{dt} E(t) < 0, \text{ otherwise}
\end{cases} \tag{7.5}
\]

Based on the calculation in equation (6.9), the derivative of the energy function is:

\[
\frac{dE}{dt} = \sum_{i=1}^{n} \frac{s_i - s_i^*}{s_i^*} \frac{ds_i}{dt} \\
= -\sum_{i=1}^{n} \frac{s_i - s_i^*}{s_i^*} l_i^* \frac{s_i^*}{l_i^*} \left[ v_x \cos \beta_i + v_y \sin \beta_i \right] \tag{7.6}
\]

Denoting

\[
\lambda_i = \frac{l_i^*}{l_i^2}, \tag{7.7}
\]

84
and combining with (7.3), equation (7.6) is simplified as

\[
\frac{dE}{dt} = - \left[ v_x^2 + v_y^2 \right] = - \sum \lambda_i^2 (s_i - s_i^*)^2 \leq 0 \quad \Box 
\] (7.8)

Since the distances \( l_i \) and \( l_i^* \) are non-negative, equation (7.8) shows that the control law of equation (7.3) is stable and converges to \( s_i = s_i^* \) (i.e. the reference home position). However, \( l_i \) and \( l_i^* \) are not directly measurable in practice, though observable. Following the error analysis in [40], the \( \lambda_i \)'s can be approximated by constants, since they do not affect the convergence. Further validation via simulation is given in section VIII.B.1).

### 7.3 Outlier rejection

When deploying the scale-based visual servoing in a real environment, it is very likely that the environment imaged at the home position will not be kept static for each homing process. For instance, the original image may contain furniture or objects that are moved over time. In general, in an omnidirectional image, these objects only cover a small portion of the image. In this section, I will describe how to remove the features on these “dynamic objects” from the set of matched features. Note that features on objects that have disappeared from the home image do not raise a significant issue because, in general, they are hardly matched to anything in the current frame.

In order to enhance the reliability of the control method against dynamic objects, I implemented a simple histogram-based rejection method. The underlying assumption is that the main transformation between the current image and the home image is simply a pure rotation. This is true as long as the features are far enough from the camera, or as long as the distance between the current position and the home position is small in comparison to the average distance to the features. In practice, I take each matched feature in the current image, and use the bearing difference with its home correspondence to vote for a potential rotation angle. Once every feature has casted its vote, the resulting histogram contains a main peak corresponding to the main rotation. Other peaks correspond to the set of features that moved coherently between the two images, i.e. dynamic objects. The background of the histogram refers to the outliers that violate the hypothesis of a pure rotational error. Mostly the corresponding points are close to the camera. The assumption may appear as a rough approximation of the general cases, especially when it is applied with a too lenient threshold. However, it provides
a reliable filter for gross outliers which would have the worse effect on the control law. In Section \[8.3\] I provide an evaluation on the effect of such an assumption. One instance of outlier rejection using the proposed selection approach is shown as Figure \[7.3.1\]

**Figure 7.3.1:** A result of the outlier rejection algorithm. The gray linkages mark the inliers of the matching; white for outliers.
Simulation results for visual homing

In this chapter, I present a number of simulation results to highlight the properties of the homing approaches. The first evaluation is to compare the performances of the different control methods. To this end, it is important to have a common metric to compare all the image-based homing methods, and characterize the influences of various assumptions and parameters. In particular, I will illustrate the effect of unknown distances to the features and the influence of feature detection noise and feature distribution. Besides, the influence of the pure-rotation assumption on the outlier rejection process will also be discussed.

8.1 Performance comparison using image-based visual homing

8.1.1 Comparison of all methods on a particular example

In the previous sections, I have discussed four different approaches and one related method by [122] to image based visual servoing:

- **BOVS**: Bearing-only visual servoing only takes the bearing to the features to compute the control (see eq. 6.16).
- **SOVS**: Scale-only visual servoing only takes into account the scale of the features to express the control, even though the bearing angles play roles in the interaction matrix (see eq. 6.15).
- **SBVS**: Scale and bearing visual servoing uses the full interaction matrix, where bearing and scale errors are integrated (see eq. 6.14).
- **SSVS**: The simplified scale-based visual servoing uses scale without requiring the pseudo-inverse computation (see eq. 7.3).
- **HSV**: As a comparison, I also implemented the algorithm by [122], which is named as HSVS in this thesis. The implementation follows...
the summarized equivalent equation:

\[
\begin{pmatrix}
v_x \\
v_y
\end{pmatrix} = \sum_i \text{sign}(s_i - s^*_i) \begin{pmatrix}
\cos \beta_i \\
\sin \beta_i
\end{pmatrix}
\]

Figure 8.1.1: Left: 45 degree view and top view of the simulated environment and trajectories of four different methods. Magenta: BOVS, Red: SOVS, Cyan: SBVS, Black: SSVS. Right: Error evaluation on the simulated environment referring to (20, 0)

Figures 8.1.1 depicts a simulated environment where visual features (blue stars) are assumed to be located on walls surrounding a ground mobile robot. The green star marks the start position of the robot, while the red filled circle is the target home position. A simulated robot uses the perceived features to implement the set of control laws presented earlier in the chapter. This means that the scale and bearing measurement are simulated from the distribution of features. The scale is computed using a simple pin-hole projection model in equation 6.4. The four visual servoing trajectories show that all control
laws converge to the home position in reasonable manner. However, the proposed SSVS method leads to a slightly straighter trajectory, whereas the BOVS method makes the biggest detour. These results, although anecdotal, give a hint about the properties, I demonstrate the statistical analysis in the next section.

Figure 8.1.2: The visual feature errors and error distance to the home position over time

The error field of the simulated environment is depicted on the right of figure 8.1.1 by taking (20,0) as the reference home position. Although the error field of the bearing error appears “flatter” than the scale error field, we could observe that combining bearing and scale error definitely help to shape the error field into a well-defined potential well.
Figure 8.1.2 provides a graphical view of the normalized error convergence using the various approaches. Note that the first row of the graph represents the bearing error in radian whereas the other rows represent the error in the scale or combined space. In each graph, the abscissa refers to the number of iterations. As expected, the methods proposed in this work (BOVS, SOVS, SBVS, SSVS) all show overall exponential convergence, fitting the justification of equation (6.2) indicated by [107]. Since the exponential convergence of HSVS is not mathematically guaranteed, the behavior of the error convergence does not reflect such characteristic.

8.1.2 Statistical comparison of the different approaches

In order to evaluate the generalisability of the observations made on a particular example, I evaluate statistically various measures on the trajectories resulting from each of the controllers. To this end, I use the environment depicted in Figure 8.1.1 and sample uniformly 2000 starting positions. For each sampled position, a new set of features is generated and the four controllers are run until reaching the convergence of the error or a maximum number of iteration. Figure 8.1.3 depicts the distribution of initial distances to the home position.

Please notice that different parameter setups, such as gain to the controller, may lead to different convergence properties. In the experiment, each controller was tuned once until reaching some satisfying convergence behavior, and then applied with the same control parameters to the 2000 starting positions.

I propose to compare the different approaches according to the following criteria:

- The success rate, i.e. the percentage of the sample position for which the algorithm leads to a convergence of the error signal. Note that this does not mean that the algorithm converges to the correct homing position.
- Final position error: for the cases where the controller makes the error converge, the final position error provides an evaluation of the accuracy of the approach. Because the controller measures the error in the scale or bearing space, it is important to verify that they converge to a low positional error.
- Number of iterations until convergence, only for the cases where convergence occurs.
8.1.3 Success rate

The success rate of each algorithm is given in the table below. The condition to determine the convergence is that the final position error is smaller than 0.1m.

<table>
<thead>
<tr>
<th>Controller</th>
<th>Success rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bearing-only visual servoing (BOVS)</td>
<td>87.70%</td>
</tr>
<tr>
<td>Scale-only visual servoing (BOVS)</td>
<td>98.90%</td>
</tr>
<tr>
<td>Scale and bearing visual servoing (SBVS)</td>
<td>98.90%</td>
</tr>
<tr>
<td>Simplified scale-only visual servoing (SSVS)</td>
<td>99.55%</td>
</tr>
</tbody>
</table>

From this table, we can draw two very simple conclusions. First, all the methods can statistically work in the simulated environment, and the simplified scale-only visual servoing (SSVS) converges with the best success rate. Second, the success rate of bearing-only visual servoing is not as high as servoing methods which take scale information into consideration. This result is interpreted as follows: the scale error can directly reflect the distance error, which, being in the same metric space as position control, helps define the convergence position uniquely. Meanwhile, when using the bearing measurements, the system needs to search for the convergence position by combining $n$ constraints ($n$ is the number of features). In some cases, the distribution of features does not provide enough constraints to clearly define...
the homing vector. In these cases, the system is blocked in a local minimum and fails to converge.

### 8.1.4 Final position error

![Figure 8.1.4: Final position errors](image)

In the ideal case, the system error should converge to zero, but in practice there is always a residual error remaining. We identify the arrival of the robot at the home position by detecting that the difference of two consecutive errors is lower than a small threshold $\sigma$.

Figure 8.1.4 shows the final position error for all the visual servoing approaches after arrival. All approaches provide acceptable convergence results. In order to explicitly show the error differences, a box-plot of the final position errors is used to describe the mean, covariance and outliers. The proposed method (SSVS) shows the lowest final position error. Although the result may be influenced by the selection of parameters, this characteristic indeed indicates that SSVS is a potential homing method with high control precision.
(a) Number of iterations. The lower figure depicted the residuals by subtracting the regression mean.

(b) Box-plot of the residuals

**Figure 8.1.5:** Number of iteration required against the initial distance to the home position
8.1.5 Number of iterations

The distribution of the number of iterations that required to reach the goal, as a function of the distance to the goal, is depicted in Figure 8.1.5. As expected, a proportional trend can be found for all approaches: the farther from the goal, the longer it takes to reach it.

The box-plot on the Figure 8.1.5(b) shows that the simplified SSVS controller and the SBVS have larger spreads than the scale-only SOVS controller. The very tight spread of the number of iterations for the compound method (SBVS) makes it a good candidate in applications where fast convergence is required.

8.1.6 Influence of feature distribution

When the feature distribution is bad (less uniform), the influence on the traditional methods is obvious, as shown in Figure 8.1.6. The main reason is that all the traditional methods depend on the inversion calculation of the interaction matrix. In extreme cases, the system gets into the neighborhood of local zeros. One typical example is the BOVS. When the features are grouped together, the zero solution to the system equation lie on a circle centered on the average feature position. Although the scale based methods are influenced as well, the distance errors, which are reflected by the scale errors, are more reliable than the bearing errors. This assertion can be observed in the error analysis in Figure 8.1.6(b): the scale-based methods such as SSVS (black) and SOVS (red) are less sensitive to the feature distributions than bearing-only (BOVS) method. Please note that the SBVS is very unstable in this case, probably due to the introduction of the bearing errors into the system matrix.

As a summary of the comparative simulation results, the simplified method can provide the lowest final error, relatively better trajectory, highest success rate and lowest computational complexity, with marginal divergence of iteration steps comparing to traditional visual servoing methods. Above all, it functions more reliably under bad feature distributions. For practical mobile robot missions, this is an important feature since the real environment usually presents complex feature distributions. To further evaluate its performance, we should remember that it is developed using a number of assumptions in section 7. For instance, I assumed that the distances to the features would not affect the convergence. In the next sections, I will evaluate the effect of these assumptions on the proposed SSVS method. At the same time, evaluate the influence of noisy observations.
8.2 Influence of the assumptions for SSVS

8.2.1 Influence of known distance assumption

In the standard visual servoing approach as well as in the simplified approach, the convergence proof relies on the knowledge of the distance to the landmarks. In practice, this assumption cannot be fulfilled with a single camera. For the standard visual servoing, it has been shown that assuming that all the landmark are at unit distance from the camera leads to an efficient control.

Because we take this assumption also for the scale-based visual servoing (SOVS and SSVS), it is necessary to show its influence on the trajectories. Figure 8.2.1 shows that there is indeed a small difference, but the performance loss is minimal, in terms of additional traveled distances.

However, it is important to note that the performance difference of the controllers is not obvious, comparing with/without distance ground-truth to features. The only difference is that the distance knowledge leads to various gains. In order to get a fair comparison, we have to tune the gain so that the numbers of required iterations are similar for both cases. After 100 simulations, an average traveled distance of 42.1 (with unknown distances) against 41.3 (with known distances) is obtained. The average absolute curvature are
also similar: 0.036 against 0.030, respectively. It indicates that the scale-based approach is not sensitive to the known distance assumptions. Therefore, the distances can be safely approximated to 1 for real implementations.

### 8.2.2 Influence of noise

The simulation of the influence of noise was run 2000 times. The added noise is distributed following a zero-mean Gaussian. The variance of the noise represents the noise level, which goes from 1m to 10m. The physical meaning is as follows: the noise level indicates the uncertainty in estimating the distance to the features, which is equivalent to the uncertainty in estimating the scale of the features. In the simulated environment, noise levels which are higher than 5m can already be considered as significant noise. One example of the influence of observation noise on the trajectory is shown in green in Figure 8.2.2. From the error plot in Figure 8.2.2(a), the bottom of the previous defined exponential well is flattened. For this reason, the robot motion converges less precisely to the home position. Also, this explains why the noise has less effect on the homing vectors of the positions which are far away from the home position.

A further analysis over the 2000 tests is given as Figure 8.2.2(c). All the three variables – extra length traveled, final position errors and iterations – rise following the increased observation noise. However, even for the maximum noise level, the controller is still stable and converging to a neighborhood of the home position.
Figure 8.2.2: Trajectory on noisy observation, and error analysis. The error analysis contains the trends of extra length traveled, final position error and iterations required over different noise levels. All the units are related to the simulation environment.
8.3 Parametrization of the outlier rejection

A successful homing process does not depend on a huge number of matched features. Therefore, a relatively low inlier ratio, e.g. 50%, is usually more than sufficient to control the robot, given reasonable large number of matches. As a reminder, I select inliers if they globally correspond to a pure rotation of the robot, thus ignoring the translational error with respect to the home position. This is done by first finding the best rotation by a voting scheme, and then selecting as inlier the features that correspond to a best rotation approximation, within a given angular threshold.

In order to evaluate within which region would such an assumption be true, I carry out the simulation as follows. Considering the simulated environment depicted in Figure 8.1.1, denoting the position at coordinate (0, 0) as the reference, I evaluate all the other positions in the simulated space (with a resolution of 0.1m) by measuring the number of features that do not fit the assumption, namely the outlier ratio.

The analysis result is shown in Figure 8.3.1. The color in Figure 8.3.1(a) implies the minimum required value of the bearing threshold (in radian) for different inlier ratio. This defines the area of the workspace where the transformation between the observed features and the reference can be seen as a pure rotation within a given threshold and for a desired inlier ration (100%, 70% and 50%). Intuitively, a lower demanded inlier ratio will relax the needs on a precise threshold. Figure 8.3.1(b) depicts an alternative visualization of the results from Figure 8.3.1(a). It shows the ratio of outliers by fixing the rotation threshold. The darker color indicates lower outlier ratio, i.e. potentially better matching results. Taking the lowest figure in Figure 8.3.1(b) as an instance, the dark area implies that by allowing a rotation threshold 1.5rad, potentially sufficient number of matches can be retrieved from a large area around the home position, though it may lead to a high false negative ratio.

For real applications, the rotation threshold needs to be tuned according to the characteristic of the feature distribution in the test environment. Empirically, a threshold of 1.0 radian is chosen, resulting in a good outlier rejection behavior while still keeping an inlier ratio of more than 80% on the data-set introduced in section 9.2.
Figure 8.3.1: Evaluation of the assumption for outlier rejection

(a) Bearing threshold (in rad) for a fixed inlier ratio
(b) Outlier ratio by fixing bearing threshold (in rad)
Chapter 9

Experiments and Discussion

9.1 Node validation

The simulation results in Figure 8.1.6 have shown that a bad distribution of features can easily ruin a trajectory. Therefore, determining whether an area is suitable or not for the visual servoing and selecting the best-fit position as potential reference point are very relevant questions for a real implementation. To address this problem, I define a quantified parameter which validates different feature distributions.

To illustrate the node evaluation metric, I considered the distributions of the features in a real case. We took an image using the omnidirectional camera, and changed the feature distribution manually, using a piece of plain paper to cover the field of view. A screen-shot of the angular feature distributions is shown in Figure 9.1.1. The bottom histograms show how many features lie in a given vertical sector of the unwrapped omnidirectional image.

The number in percent is the positive matching ratio, the integer in squared brackets is the maximum number of matching points within a single bin in the histogram; after that is the number of matching key-points; at the end is the score $S$ of the distribution. The score $S$ is calculated from the entropy of the distribution of the positive matches, by

$$S = - \sum p_i \ln p_i$$  \hspace{1cm} (9.1)

where $p_i$ is the ratio of the matched keypoints in each bin of the whole histogram. Our method inherently assumes that a more equalized distribution will lead to better performance of the homing ability. In the simulations (see Figure 8.1.6), I show a situation where SSVS is less sensitive to distribution changes than other methods. For general purposes, a quantified factor that assesses how well an environment is suited for the homing task can be useful.

The maximum score $S$ can be calculated from a uniformly distributed histogram. Empirically, if $S$ is greater than 80% of this maximum score,
the reference home position is acceptable. It is worth noticing that the number of matched features (82) is still high in Figure 9.1.1(b). Typically, in a less textured indoor environment, the homing method can work with a minimum number of positive matches around 40. This indicates that the number of positive matches can not fully indicate the potential of whether an environment is fit for visual homing task or not.

\[ S = \ln 256 = 5.55 \] therefore the score threshold is 4.44.
## 9.2 Results on an indoor data-set

<table>
<thead>
<tr>
<th>Database</th>
<th>BOVS</th>
<th>SOVS</th>
<th>SBVS</th>
<th>HSVS</th>
<th>SSVS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TAAE</td>
<td>Max</td>
<td>Min</td>
<td>StdVar</td>
<td>TAAE</td>
</tr>
<tr>
<td>A1originalH</td>
<td>15.16</td>
<td>50.07</td>
<td>6.52</td>
<td>6.50</td>
<td>16.04</td>
</tr>
<tr>
<td>Hall1*</td>
<td>12.97</td>
<td>29.55</td>
<td>6.95</td>
<td>4.67</td>
<td>23.18</td>
</tr>
<tr>
<td>Hall2*</td>
<td>21.04</td>
<td>42.03</td>
<td>14.02</td>
<td>4.55</td>
<td>34.28</td>
</tr>
<tr>
<td>Roeben1H</td>
<td>27.68</td>
<td>48.50</td>
<td>15.17</td>
<td>5.89</td>
<td>25.86</td>
</tr>
<tr>
<td>CHall1H</td>
<td>12.21</td>
<td>28.09</td>
<td>6.39</td>
<td>3.96</td>
<td>22.34</td>
</tr>
<tr>
<td>CHall2H</td>
<td>18.45</td>
<td>47.63</td>
<td>10.69</td>
<td>5.70</td>
<td>34.16</td>
</tr>
<tr>
<td>Chair*</td>
<td>17.66</td>
<td>29.10</td>
<td>8.77</td>
<td>4.15</td>
<td>24.71</td>
</tr>
</tbody>
</table>

Table 9.2.1: Error Analysis for all the algorithms (in degree)

In order to compare all related methods under the same conditions, I tested on a widely cited data-set provided by Vardy [126]. The data-sets are a collection of panoramic images (some with unwrapped version) obtained from typical indoor environments, plus the calibration information and pose information. Some of the data-sets include dynamic objects in the scene, such as a mobile wheeled chair etc. All the raw images are with 640x480 resolution (unwrapped image: 561x81), the actual intervals between two nearest nodes are equal constants, typically 30cm. By taking position (5,8) of A1originalH data-set as the reference home, the homing vectors calculated from other images in the same data-set using different methods are shown as Fig. 9.2.1(a). The color of the filled circles indicates the differences in number of matched features. It is interesting to see that SSVS exhibits clean behavior pointing towards the home position, even when the matching ratio is low.
Figure 9.2.1: Homing vectors and error analysis referring to (5.8). The color-map in the first row indicates the number of matched key-points; in the second row, the color-map indicates the average angular error in degree.
According to the comparison made in [122], the TAAE (total average angular error) can be an important statistic result when evaluating the homing ability. The overall AAE (average angular error) can be obtained as follows:

\[ AAE(ss) = \frac{1}{mn} \sum_{x=1}^{m} \sum_{y=1}^{n} AE(ss, cv_{xy}) \]

where \( AE \) is the absolute angular error between the computed homing vector and the ground truth. The subscript \( ss \) and \( cv \) stands for the reference scene and current views respectively. For the entire image database \( db \), the TAAE computes the overall average of \( AAE(ss) \) as follows:

\[ TAAE(db) = \frac{1}{mn} \sum_{x=1}^{m} \sum_{y=1}^{n} AAE(ss_{xy}) \]

The AAE for each position of A1originalH is illustrated in Fig. 9.2.1b).

Test results on the full Vardy data-sets are illustrated in table 9.2.1 where I show an extended statistics of the angular error, such as maximums, minimums and standard derivations. Darker background on specific numbers mark the best performance of the row. The star markers after the name of data-sets indicate that those data-sets are unwrapped by detection and operation on the largest circle of the panoramic images. In terms of precision, SSVS and BOVS show the best performance in general. However, we must notice that BOVS requires a good enough estimation of the global heading, which is in general vague using visual compass methods [125]. Moreover, since SSVS and HSVS hold the lowest computational complexity, the advantage of SSVS is revealed.

The generic statistical result of the test is shown in Figure 9.2.2. The blue curve indicates the processing time over 28900 matching operations (cross matching of 17*10 image array). It shows that the processing frequency is in average around 2Hz to 7Hz, depending on the resolution level. In this experiment, I use four different resolution levels for the input images. Each resolution level corresponds to the starting octave level for the image pre-processing as well as the extraction of the SIFT features. Generally speaking, a resolution level of 0 represents the original size of the image, -1 means an upsample of the original image to the double of its size. By this mean, I control the amount of information processed by the algorithm. This illustrates the influence of the number of features on the algorithm. An image with better resolution (e.g. resolution level -2) can provide more features since it has more pixels and more details, but it will require more processing time. The
result shown in Figure 9.2.2 provides also a baseline, which can be used to evaluate whether the algorithm is applicable for real-time tasks. Note that more than 99% of the computation time is for the SIFT extraction and feature matching.

Taking the simplified approach as an instance, TAAE is also plotted under different resolution level with error bars, using the purple curve in Figure 9.2.2.

Figure 9.2.2: Average Error and processing time with different resolution levels

A detailed analysis on the effect of different resolution levels on AAE, and the computation time is shown in table 9.2.2. It shows that at a higher resolution (at a lower resolution level), all methods can work better in general. A primary reason is that the higher resolution leads to more feature points, which provides more constraints for the error correction. Note that the simplified method can provide faster and more consistent results than others. The bearing-only visual servoing approach provides best results at node (5,8), in the sense of low AAE, at the cost of 4 to 10 times more computation time than the simplified method. However, this result does not prove that the bearing-only method is the best for all the nodes: according to further tests, the TAAE of the bearing-only method is 11.53 degrees, with a bigger variance than the simplified approach. In this thesis, I chose to focus on the simplified
approach for its lower computation requirements.

Table 9.2.2: Effect of image resolution on the AAE referring to (5,8).

<table>
<thead>
<tr>
<th>Resol.</th>
<th>Metric</th>
<th>BOVS</th>
<th>SOVS</th>
<th>SBVS</th>
<th>SSVS</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2</td>
<td>AAE</td>
<td>6.37</td>
<td>10.48</td>
<td>13.87</td>
<td>7.75</td>
</tr>
<tr>
<td></td>
<td>Time(µs)</td>
<td>25531</td>
<td>21579</td>
<td>27584</td>
<td>6528</td>
</tr>
<tr>
<td></td>
<td>AAE no OR</td>
<td>9.88</td>
<td>10.67</td>
<td>14.18</td>
<td>8.54</td>
</tr>
<tr>
<td></td>
<td>Time(µs)</td>
<td>25424</td>
<td>20260</td>
<td>30920</td>
<td>8162</td>
</tr>
<tr>
<td>-1</td>
<td>AAE</td>
<td>6.36</td>
<td>11.86</td>
<td>13.42</td>
<td>8.13</td>
</tr>
<tr>
<td></td>
<td>Time(µs)</td>
<td>18421</td>
<td>14594</td>
<td>20459</td>
<td>4804</td>
</tr>
<tr>
<td></td>
<td>AAE no OR</td>
<td>14.52</td>
<td>12.52</td>
<td>15.96</td>
<td>8.04</td>
</tr>
<tr>
<td></td>
<td>Time(µs)</td>
<td>24584</td>
<td>15974</td>
<td>22730</td>
<td>5384</td>
</tr>
<tr>
<td>0</td>
<td>AAE</td>
<td>6.26</td>
<td>12.74</td>
<td>14.69</td>
<td>9.90</td>
</tr>
<tr>
<td></td>
<td>Time(µs)</td>
<td>18017</td>
<td>9591</td>
<td>12303</td>
<td>2464</td>
</tr>
<tr>
<td></td>
<td>AAE no OR</td>
<td>24.67</td>
<td>16.16</td>
<td>19.06</td>
<td>10.14</td>
</tr>
<tr>
<td></td>
<td>Time(µs)</td>
<td>18358</td>
<td>10456</td>
<td>13220</td>
<td>2860</td>
</tr>
<tr>
<td>1</td>
<td>AAE</td>
<td>8.39</td>
<td>16.37</td>
<td>17.86</td>
<td>12.24</td>
</tr>
<tr>
<td></td>
<td>Time(µs)</td>
<td>11044</td>
<td>5875</td>
<td>6231</td>
<td>1037</td>
</tr>
<tr>
<td></td>
<td>AAE no OR</td>
<td>12.86</td>
<td>18.25</td>
<td>20.00</td>
<td>13.23</td>
</tr>
<tr>
<td></td>
<td>Time(µs)</td>
<td>14124</td>
<td>6792</td>
<td>7046</td>
<td>1254</td>
</tr>
</tbody>
</table>

All AAE values are in degrees. “AAE no OR” stands for AAE without outlier rejection.

In this implementation, I used images with resolution level -1, which leads to nearly 4.0 Hz processing frequency. Practically it is sufficient for the real-time applications under consideration. The $TAAE$ of the simplified approach ($12.71^\circ$), over the entire database, outperforms scale space homing method ($15.79^\circ$) [122] and warping method (snapshot model) ($46.6^\circ$) [109].

Finally, for different positions in the dataset, we can observe different stability behaviors of the homing vectors. We interpret this as a result of the variation of feature distributions. It is also an important reason why we need to setup a metric to evaluate nodes with respect to their feature distributions (see Section 9.1). To confirm this assumption, I run an evaluation on the dataset used earlier [126]. Figure 9.2.3 illustrates the correlation between a low AAE value – darker shades in (b) – and a greater entropy of the positive matches (a).

---

2Extended information: TAAE of other methods are: bearing-only($11.53^\circ$), scale-only($13.51^\circ$), compound method($15.57^\circ$).
9.3 Results of real-time homing test

9.3.1 Homing vectors to one reference

In order to show the performances in a more dynamic environment, a dataset taken together with Vicon motion capture system is used for further evaluation. In this test, four people are continuously moving in the scene where the robot is taking the data-set. A sample image is shown in Fig. 9.3.1.

![Sample image of the data-set with arbitrarily moving objects](image)

Figure 9.3.1: Sample image of the data-set with arbitrarily moving objects

The calculated homing vectors taking (0,0) position as the reference home is depicted in Fig. 9.3.2 using SSVS. We could see that the robot trajectory is arbitrary. Thanks to the motion tracking system, the 6 DoF ground truth is simultaneously recorded. It can be noticed that even with relatively low
number of matched feature points, the robot can show reasonable homing directions using SSVS.

![Colorbar indicates the number of matched points](image)

**Figure 9.3.2:** Results of SSVS with moving people in the scene

As comparison, only the outperforming algorithms from previous tests – BOVS, SSVS and HSVS are carried out on this data-set. Concerning BOVS requires global heading estimation, I carry out the evaluation for BOVS using the ground truth heading information. In practice, visual compass is required to provide such information, leading to worse performance. For related works of visual compass, the readers are referred to [125]. The histograms of the angular errors for the three methods are shown in Fig. 9.3.3. It indicates that the SSVS has the best precision, and BOVS performs the worst even with ground truth heading information. This result shows SSVS is better fit for robot navigation tasks in dynamic environments for real-time applications.

### 9.3.2 Experiment of outlier rejection

The outlier rejection method is the key to ensure the correctness of generated homing vectors in dynamic environments. We also compare the cases of with
Figure 9.3.3: Histogram of angular errors for the selected algorithms under the test condition with moving people
and without the proposed outlier rejection method. The statistics of the generated angular errors show the effect of the proposed outlier rejection method, as depicted in Fig. 9.3.4. It shows that the implementations with the proposed outlier rejection have lower error mean and smaller derivation. Regarding the simple assumption, the additional computation required for outlier rejection is minor. Therefore, such an algorithm is generally feasible for all similar applications using key-point features obtained from omnidirectional cameras.

Figure 9.3.4: Error comparison in the cases of with/without outlier rejection. The labels marked with _on indicate the boxplot is with outlier rejection, _off indicates the results calculated from the raw matching result.
In this section, I present a novel integrated indoor topological navigation framework, which combines odometry motion with visual homing algorithms. I show the robustness to scene variation and real-time performance through a series of tests conducted in four real apartments and several typical indoor scenes, including doorways, offices etc.

10.1 Overview

The proposed navigation system is built on a topological graph, with an example shown in the left of Figure 10.1.1. Each node on the graph represents a pose (or navigation waypoint) in the environment. The system is modeled as a state machine structure, whereby the navigation and mapping processes can be switched. The transition between two neighbouring connected waypoints is performed by a two-phase algorithm using odometry and visual servoing. These two phases are denoted by general positioning and pose stabilization.

At the system level, the navigation system is structured as shown in Figure...
10.1.2 Configuration layers such as sensors, data and low level processes are shown in different colors. The omnidirectional camera is used as the main sensor for navigation, while a 3D stereo vision system is used for local obstacle avoidance.

Due to the multi-layered system structure, an efficient management mechanism needs to be defined. A state-machine was built up as shown in Figure 10.1.3. It can be divided into two major parts: mapping and navigation.

### 10.2 Mapping

The first issue regarding the mapping problem is how to represent the map and descriptors for topological nodes. Using a typical topological map, the high-level structure is a simple graph, which uses the vertices to represent the nodes and edges to represent the translation relationships between nodes. This is shown in Figure 10.1.2. For each node, a structure containing keypoint
features and odometry differences to neighbors is created. Specifically, the Mapping state in Figure 10.1.3 represents the mapping process, which requires basic states such as moving and freezing of the robot. When the robot arrives at the reference position, a node will be created upon the request from the user. The node structure will be registered locally and also saved in a remote database. The Mapping state can always be triggered whenever the robot is in the Ready state. It is a useful feature because it allows extending the topological map from any node, regardless whether or not the node was newly created, e.g. the mapping process can also start from any intermediate node occurring on the route. As shown in Figure 10.1.1, a global map can be easily extended to several branches and dynamically managed.

### 10.3 Localization in the map

The localization in a topological map means that the robot can name its current location as the nearest node. The performance of the localization will greatly affect the navigation in the map. Therefore, I implemented a localization method that relies on visual feature matching.

Comparing the current set of feature points $F_c$ with each feature set
\( F_i, i \in [1, n] \) in the database, we search for the best \( F_{\text{best}} \) and second best \( F_{\text{second}} \) matching nodes, according to the matching ratio \( r_{\text{best}}, r_{\text{second}} \). To ensure positive matching, we require

\[
\frac{r_{\text{second}}}{r_{\text{best}}} \leq 70\%
\]

When this constraint is met, the current node is exclusively located. Otherwise, the localization will return several possibilities which have to be verified at a later time. At extreme cases, the robot might be incorrectly localized, causing navigation process based on the localization to fail. The localization process can be triggered again at the failure poses.

10.4 Navigation

A global planner based on Dijkstra algorithm will perform path planning over all topological nodes in the graph. It results in a sequence of nodes which the robot needs to follow in order to navigate from the current position to the target position. As such, the topological navigation is decomposed into several node-to-node phases. Each node-to-node action is performed through the Navigation state shown in Figure 10.1.3.

The Alignment state means the robot attempts to align its orientation (heading direction) with the pose saved in the database. The translation between two nodes is handled by a two-phase method: general positioning (state=General positioning) and pose stabilization (state=Pose stabilization). The former is a combination of obstacle avoidance and odometry based position control. The latter is used to correct the accumulated odometry error for pose stabilization.

The procedure of general positioning is shown in Figure 10.4.1. Two exceptions during the positioning process are “Target arrived” and “Obstacle detected”. When obstacle(s) are detected, the system will switch to an exclusive obstacle avoidance control process. The mechanism ensures that when the robot’s path is not blocked. In particular, the obstacle avoidance mode will take over the motion control using a naive obstacle contour following rule. I set the time span for obstacle avoidance mode to six seconds for experiments in a typical apartment environment. After the obstacle avoidance process, the robot will try to reach the target again using the “Heading to the Target” process, which is based on the odometry difference between current position and the target position.

As for the finishing conditions, there are two possible states during general positioning. They both can trigger the finish of this procedure. During the
process of general positioning, once enough positively matched keypoints are observed, the working state will switch to pose stabilization automatically. This means the current position is near enough for a robust pose stabilization process. Second, when the odometry of the current position matches the target odometry within a bounded error region, the state-machine will also switch to the pose stabilization state. The pose stabilization is structured under the IBVS framework as shown in Figure 10.4.2. It uses an omnidirectional camera as the only sensor, and tries to correct the error in the image space by controlling the robot motion. It is able to perform a visual homing algorithm in real-time, by which the accumulated odometry error will be corrected. The algorithm retrieves the target image and features from the database and matches the current image with the target image. These image features are fed into the core visual servoing algorithm to generate homing vectors which direct to the target waypoints.

### 10.5 Visual Compass

One subtle problem is that both for general positioning and pose stabilization the estimation of heading is important. Here I propose a visual compass
solution for the orientation estimation. An important consideration is that a visual compass is not affected by disturbances generated by high currents, armouring irons etc.

Considering related works, e.g. [125] proposed a summary of existing visual compass algorithms, we can see weighted average is often used. This idea can be extended by only using the features which have less change in their scale size. Specifically, each feature $p_i$ is assigned a weight $w_i$:

$$w_i = e^{-k \cdot ds_i}$$  \hspace{1cm} (10.1)

where $k$ is a constant parameter and $ds_i = s_i - s_i^*$. The heading difference $\phi$ is then calculated from a weighted average of the bearing differences $d\beta_i = \beta_i - \beta_i^*$, as

$$\phi = \frac{\sum d\beta_i \cdot w_i}{\sum w_i}$$  \hspace{1cm} (10.2)

A qualitative result comparing the proposed visual compass and magnetic compass is shown in Figure 10.5.1.
10.6 Experiments and results

The navigation system has been tested at several indoor environments with different setups. Figure 10.6.1 shows a collection of robot images and a typical sketch of robot structure.

As shown in Figure 10.6.1(c), a pan-tilt unit is mounted to hold the stereo camera system. The motion of the pan-tilt unit will follow the control speed of the robot, in order to predict and observe potential obstacles. The detected local obstacles will be registered onto a local navigation cost-map. The cost-map is then used to constrain the trajectory of the robot.

A screen-shot of the created topological map is shown in Figure 10.6.2. It depicted the general surveillance GUI. I also designed a simple control panel for the mapping and navigation task. It links to a PostgreSQL database, which is used to manage (store/retrival) the topological map and corresponding features, as shown in figure 10.6.3. The organization map of the conducted relation database is shown as Figure 10.6.4. It depicts that the database is centered on Public relation tables, which describe the relations among nodes and features. Whenever a new node is registered on the map, the displacement relation table and tags are updated. All related sensor informa-

---

Cost-map implementation uses ROS package [www.ros.org/wiki/costmap_2d](http://www.ros.org/wiki/costmap_2d)
Figure 10.6.1: Robots collections.

Figure 10.6.2: Example of map with links between the topological nodes (yellow), the robot (magenta), the data from stereo system (white).
Figure 10.6.3: The control panel used for topological mapping and navigation.

...tion such as the image taken at the node position, visual features and laser information (for further reference) are recorded.

The experiment has been carried out separately both in day and night time conditions in order to test the robustness to light changes.

10.7 Functional test

An experiment of around 20min depicted by a plot of the visual servoing error is shown in Figure 10.7.1. In order to get a more compact result, the logging for this plot will only be triggered when the system is switched to pose stabilization mode. It means that every time the error drops below the green bar (which marks the threshold of the accuracy), the system switches to general positioning state. It is important to note that there are time spans after each mode switching, which are not visible to the readers from the plot. The figure depicts that the visual error can converge for every control cycle. An issue should be mentioned, regarding the areas marked with brown rectangles. In these areas, unstable convergences can be observed. After the analysis of the log and experiment process, the reason for this kind of behavior can be identified as follows. Our testing robot was a differential driven robot with non-omnidirectional dynamics. Therefore, the constraints in the motion model implies that the robot motion can reach singularities. This occurs in the cases when the extension of the straight line between the two active wheels passes through the target position. To solve this problem, I propose the following control law. The homing vector is translated into...
Figure 10.6.4: Relational Database Organization
Figure 10.7.1: Pose stabilization error vs measurement indices. The red rectangle is a magnified view of the errors in the corresponding region. The vertical axis is in the unit of visual servoing error.

A linear speed $v_L$ and a rotation speed $\omega$, using (10.3). $K_\alpha$ and $k_\alpha$ are two parameters defining respectively the rotation speed gain and the trade-off between movement speed and heading alignment.

\[
\alpha = \arctan 2(v_y, v_x) \\
v_L = e^{-k_\alpha \alpha^2} \cdot \| v_x \| v_y \| \\
\omega = K_\alpha \cdot \alpha
\]  

(10.3)

10.8 Longer time roaming test

In order to test the visual navigation approach over longer time, a simple script was used to generate a sequence of random target nodes covering the whole transitional space. From a full battery status to empty, the robot drove around for 1 hour, successfully reaching more than 40 different targets. The total graph, shown in Figure 10.1.1, contains 18 topological nodes in a 3 bed-room apartment including kitchen and living room ($130m^2$ in all). During the test, four people were randomly walking around the apartment. This confirms the reliability of the approach. For more intuitive result, please refer to the video [127].

The existing test results in real apartment environment showed the robustness of the method against some illumination changes and occurrence...
of dynamic objects in the scene. It also shows that the method worked well at the nodes with richly textured scenes and unstable in low textured ones. It means that a memory mechanism combining working memory and long-term knowledge is necessary for a long-term navigation application.
In this part, I presented a visual homing framework using visual servoing, based on the scale and/or bearing measurement of popular visual features such as SIFT and SURF. After showing how these measurements could be used in the standard visual servoing framework, I proposed a simplified controller with a complexity linear with the number of observed features. I have demonstrated the usability of scale-based visual servoing and I have shown that the simplified approach is stable and offer similar performances as the fully-fledged scale-based visual servoing. When tested against a standard data-set, the proposed controller outperformed state-of-the-art approaches in terms of average angular error.

Overall, the scale-based approach shows a certain robustness to dynamic changes in the environment. Similarly to any visual-based homing solution, the proposed approach is quite sensitive to lighting changes, even if the research team has been able to reuse a map during the day-time, which was acquired during the night. Based on these results in real apartments and office environments, the combination of scale-based visual homing and odometry has shown to be a successful lightweight algorithm for indoor navigation using a single camera as main sensor.
What is the good of drawing conclusions from experience? I don’t deny we sometimes draw the right conclusions, but don’t we just as often draw the wrong ones?

G. C. Lichtenberg (1742-1799)
Chapter 12

Conclusion

12.1 Summary

The goal of perception process is to enable robots to obtain information from an environment. Through perception, it is essential for robots and people to be able to share a common understanding of the environment. According to psychological research, humans use mostly topological representations to describe their surroundings. Therefore, topological environment modeling for mobile robot is the key for mobile robots to work with people in a cooperative manner. This thesis aims at solving the fundamental problems of topological modeling of an environment for mobile robots using an omnidirectional camera as the only sensor. Inspired by the pattern in which humans recognize the world and perform navigation, I split the topological modeling problem into two aspects. Firstly, scene clustering and recognition need to be fulfilled online. It means that different scenes need to be clustered. Meanwhile, when the robot has returned to a previously visited place, the scene needs to be recognized. Secondly, topological visual navigation among way-points needs to be solved in real-time. In this thesis, the former was tackled by combining a Simultaneous clustering and recognition (SCAR) framework with a lightweight color-based descriptor - Fast Adaptive Color Tags (FACT); the latter was solved by integrating visual homing with a state-machine based control scheme. Figure 12.1.1 depicts the technical structure of topological modeling. Regarding the two aspects, this dissertation was organized as two parts respectively.

I started with casting visual solutions of the topological mapping to a SCAR problem, using an omnidirectional camera (Part I). In chapter 1, I presented DP-Fusion, which is a novel on-line information fusion framework for multi information based on Dirichlet Process Mixture Model (DPMM). It combined synchronized sensor readings to automatically cluster data into models, while recognizing data from existing models simultaneously. Results showed its advantages as its on-line computing mode and low computational
cost. It also implied that the inference of a DPMM can be approximated by the product of the conditional probability distributions. It can be envisioned that the proposed \textit{DP-Fusion} framework can easily generalize to SCAR applications using synchronized multi information. This framework was taken as the basis for the topological scene segmentation problem.

Before getting into details of scene recognition problem, an advanced descriptor for the sensor measurements is required. In Chapter 2, a descriptor for omnidirectional image, named \textit{FACT}, was proposed, which used color information and geometric information extracted from unwrapped panoramic images. On one hand, it perfectly fits the assumptions of \textit{DP-Fusion}, since both information are extracted from image frames and readily synchronized; on the other hand, its robustness was demonstrated by evaluations in a real environment using a naive matching algorithm. However, the naive matching algorithm induced several drawbacks, such as false positive ratio of scene changing detection is high, and a broad parameter-set needs to be tuned carefully. These defects were especially taken care of, when the proposed \textit{DP-Fusion} framework was applied, leading to the scene clustering and recognition using \textit{DP-FACT}.

\textit{DP-FACT} was introduced in Chapter 3. As a special case of \textit{DP-Fusion}, it used the two synchronized information embedded in \textit{FACT}. The observations were converted to histograms in order to introduce statistical meanings and
to facilitate the matching operation. Each histogram was compared with existing models during the exploration. As the existing models are in the format of histograms as well, the matching of two histograms was easily achieved by $\chi^2$ tests. In order to keep the models up-to-date, an incremental update was applied. By far, a pipeline from descriptor extraction to model generation and validation was implemented. The experiments using this pipeline were provided in Chapter 4.

In Chapter 4, I compared the proposed $DP$-$FACT$ with key-point based approaches such as SIFT and BRISK. I show that the proposed approach is more stable in scene change detection and recognition simultaneously. It also validates that the inference of a DPMM can be approximated by reasoning the conditional probability directly using $DP$-$Fusion$ framework. Note that the proposed FACT descriptor only dealt with indoor environments, where vertical lines are preserved in the field of view of unwrapped panoramic images obtained by omnidirectional cameras. Therefore, the results did not imply that the extended applications for semi-structured environment are easily feasible. Notwithstanding this limitation, it did suggest that color based features can be integrated with a real-time online scene recognition and topological mapping robotics system, with relatively good performance. We can imagine the combination of keypoint- and color-based methods will help to solve this problem at a hybrid level, without limiting the targeting environment. Regarding loop-closing problem, the proposed framework can help in the selection of target poses to be matched, with low computational cost. For example, the loop-closing can be simply performed to the poses in the scenes which have the same label as the current image frame. These topics can be considered as further directions for related research.

When a topological representation of the environment is maintained, it is important to allow the mobile robot to navigate autonomously among these regions. To this end, a visual topological navigation framework which is centric on visual homing was presented in part II of this thesis. This part can be separately applied for other scenarios independently. As a inspiration, the classical visual servoing framework was studied in Chapter 6. A group of novel control laws for omnidirectional camera was introduced following the IBVS schema. These control laws, upon the features they use, are classified as different types, i.e. BOVS, SOVS and SBVS etc. However, the complexity of these classical IBVS is high. In order to reduce the complexity, I derived the hints to design a simplified control law from the observability analysis of the visual system, as described in Chapter 7. The major conclusion was that the robot heading, distances to feature points are fully observable, as long as the scale variance and control command are known. Although not
explicitly calculated, the robot poses could also be retrieved using traditional structure-from-motion methods. These observations imply that a simplified control law can be formulated using scale information of observed features in a more direct manner.

The simplified control law was proposed in Chapter 8, denoted as SSVS. Started from the one-dimensional case, the control law was formed as a linear combination of weighted scale errors. Lyaponov stability was proved for the two-dimensional case, since the only equilibrium is unique where all feature scales need to be consistent for all the matched key-points. By integrating the proposed SSVS within a feedback control-loop, a visual homing solution was proposed. I also derived an outlier rejection algorithm, inspired by RANSAC, which implied the assumption that the robot motion in the neighborhood of the reference position can be considered as a pure rotation. This assumption and the performance of SSVS were further justified by simulation and real experiment.

Chapter 9 showed the simulation results of all related visual homing algorithms. Besides the newly proposed methods, HSVS was taken for comparison. As a summary of the comparative simulation results, SSVS can provide the lowest final error, relatively better trajectory, highest success rate and the lowest computational complexity, with marginal divergence of iteration steps comparing to other visual servoing methods. Above all, it was the most reliable under bad feature distributions. For practical mobile robot missions, this is an important property since the real environment usually derives complex feature distributions. Several aspects regarding the assumptions for SSVS were specifically evaluated as well. In order to demonstrate the performance of the proposed visual homing algorithms, real experiments were carried out in Chapter 10.

Because the simulation results showed that the feature distribution may affect the performance, in chapter 10, a node validation criterion was firstly proposed. It used entropy of the feature distribution to show the feasibility of a place to be taken as a reference position. After that, the test and error analysis on a group of widely cited datasets were presented. In terms of precision, SSVS and BOVS showed the best performance in general. However, we must notice that BOVS requires a good enough estimation of the global heading, which is in general vague using visual compass methods. Moreover, since SSVS and HSVS hold the lowest computational complexity, the advantage of SSVS was revealed. Finally the results of actual experiment using arbitrary walk of the robot indicated the advantage of SSVS in real scenarios.

Since the performance of the proposed visual homing algorithms is validated, a navigation framework is required in order to realize real-time appli-
cations. The navigation framework presented in Chapter 11 combined odometry motions with visual homing algorithms. At the same time, database management and necessary techniques such as visual compass were also included. The lightweight framework was organized as a finite state machine, where mapping and navigation processes are arranged in a flexible way. Test results in real apartments and typical indoor environments confirmed the robustness of the proposed framework.

As a summary, this thesis presented several aspects for visual topological modeling of the environment using an uncalibrated omnidirectional camera. Conducted simulations and experiments highlighted the new contributions derived from the novel algorithms and implementations.

12.2 Outlook

Even if experiments and simulations in the thesis showed promising results, there are still some further possibilities and extensions to be developed and improved.

The proposed $DP$-Fusion framework takes only synchronized information as input. One possible extension is to generalize it to asynchronized data. It is important for the cases where the sensor readings are with time-delay. For example, in multi robot scenarios, sensor readings from different robots can be synthesised in a unified SCAR framework.

The proposed scene recognition used a central omnidirectional camera with a mirror mounted perpendicularly to the motion plane. This the basis for the proposed FACT descriptor. However, when the motion of the robot is less constrained, the assumption that all the major vertical lines are preserved in the field of view is invalid. In these cases, fusion of extra visual hints are required. For example, keypoint-based descriptors can be adopted to form a more robust recognition approach.

In order to expand the application of the proposed homing algorithm, part of the ongoing work is to apply such control scheme to all-terrain mobile robots. Further experiments for navigation on a 2D manifold embedded in 3D terrain, such as semi-structured environments, are to be carried out. The use of feature scales can be extensively studied. E.g. the features with less scale changes should be only used for visual compass; those with major changes should be used for homing vector generation. Such optimization needs to be further studied.
12.3 Closing comments

All in all, this thesis shows that mobile robots can achieve efficient topological modeling of a typical environment using an omnidirectional camera. Comparing with non-topological methods, this thesis provides practical and low-cost solutions to scene recognition and visual navigation problems, leading to common understanding of an environment for both humans and robots. Hopefully, the readers can be inspired with new ideas in similar research directions.
Bibliography
Bibliography


Curriculum Vitae

Ming Liu was born in Zibo, Shandong, P.R.China on March 30th, 1984. He is an IEEE Student member since 2012. He received the B.A. degree in Automation at Tongji University in 2005. During his master study at Tongji University, he stayed one year in Erlangen-Nünberg University and Fraunhofer Institute IISB, Germany, as a visiting scholar. He was admitted as a PhD student directly at Tongji University without requirement of master degree under special grant in 2007, regarding his academic performance. Since 2009, he has been working as a PhD student in the mechanical department of ETH Zürich. He is a founding member of Shanghai SWing Automation Ltd. Co. He is also involved in several NSF projects, and National 863-Hi-Tech-Plan projects in China. As a team member, he won the second place of EMAV’09. He won the best student paper award as first author for MFI 2012, the best paper finalist for MFI 2012 as second author, the best paper award in information for ICIA 2013 as first author and best paper award finalists as co-author, the best RoboCup paper award for IROS 2013.

Ming Liu’s research interests include dynamic environment modeling, 3D mapping, machine learning and visual control. He’s particularly interested in the investigation of novel, real-time online approaches in solving mobile robot mapping and navigation. In general, he can be stimulated by research in all robotic and intelligent system fields. Ming Liu has been involved in the following projects:

- 09/2011~present Realize Project, coordinating and contact person. A set of joint projects between China universities and ETH Zürich.
- 02/2010~present NIFTi Natural Human-Robot Cooperation in Dynamic Environments, EU Project FP7 #247870
- 11/2008~05/2010 Robots@home STREP EU Project IST-6-045350, on Navigation for mobile robots
- 06/2008~12/2011 Environment perception and understanding for mobile robot, the National high technology Research and Development Program of China (863 Program) 2009AA04Z213
- 03/2007~06/2007 S1X Concept car design, project assistant at Siemens VDO Headquarter, Regensburg, Germany
Award

EMAV 2009

On September 17th, 2009, a team of five ASL members (Davide Scaramuzza, Stephan Weiss, Laurent Kneip, Ming Liu, and Daniel Eberli) participated in the annual European Micro Aerial Vehicle Competition (www.emav09.org), which regularly takes place in Delft (Netherlands). The competition consisted in having a small helicopter entering and exiting from small apartment in a fully autonomous manner. Our team, named ETH-Maverick, was the only autonomous team over the 8 participating teams! Furthermore, it was the only team using only a single camera to control the helicopter. Conversely all the other teams used remote control. The autonomy of the helicopter and the use of solely vision impressed the jury and gave our team the 2nd place. We missed the 1st place just because the size of the helicopter was a penalizing factor according to the rule used in the competition.

MFI 2012

On September 15th, 2012, I won two awards from the leading conference of multi information fusion: 2012 IEEE International Conference on Multisensor Fusion and Information Integration (MFI 2012). The paper “DP-Fusion: A generic framework for online multi sensor recognition” has won the Best Student Paper Award, and "Towards Real-time Multi sensor Information Retrieval in Cloud Robotic System" is awarded with Best Paper Finalist.

7th Chunhui Cup 2012

I was awarded Winning Prize for the 7th Chunhui Cup Pioneering Competition for Overseas Chinese Scholars (2012), hosted by the Ministry of Science and Technology and Ministry of Education in China, for the proposal entitled “Autonomous Agriculture Machinery: Design, Integration and Optimization”.

ICIA 2013

master student won the Best Paper Finalist by the paper “An Experimental Evaluation of the RT-WMP Routing Protocol in an Indoor Environment”.

8th Chunhui Cup 2013

I was awarded Winning Prize for the 8th Chunhui Cup Pioneering Competition for Overseas Chinese Scholars (2013), hosted by the Ministry of Science and Technology and Ministry of Education in China, for the proposal entitled “Elderly Care Household Robots”.

IROS 2013

On November 6th, 2013, as a co-author, I won the Best RoboCup Paper Award from the top robotic conference: 2013 IEEE International Conference on Intelligent Robots and Systems (IROS 2013), for the paper entitled “3D Path Planning and Execution for Search and Rescue Ground Robots”.

List of Publications

Journals


Peer-reviewed Proceedings


- Ming Liu, Cedric Pradalier, Roland Siegwart and Qijun Chen, A Bearing-only 2D/3D-homing method under a visual servoing framework, in Proceedings of the IEEE International Conference on Robotics and Automation, Alaska, the USA, ICRA 2010.


- Ming Liu, Roland Siegwart, DP-FACT: Towards topological mapping and scene recognition with color for omnidirectional camera, in Proceedings of the IEEE International Conference on Robotics and Automation, St-Paul, the USA, ICRA 2012.

- Ming Liu, Cedric Pradalier, Francois Pomerleau, Roland Siegwart, Scale-only Visual Homing from an Omnidirectional Camera, in Proceedings of the IEEE International Conference on Robotics and Automation, St-Paul, the USA, ICRA 2012.


- Ming Liu, Lujia Wang, Roland Siegwart, DP-Fusion: A generic framework for online multi sensor recognition, in Proceedings of the IEEE International Conference on Multisensor Fusion and Information Integration,
MFI 2012, **Best Student Paper Award.**


- Geert-Jan Kruijff, Miroslav Janicek, Shanker Keshavdas, Benoit Larochelle, Hendrik Zender, Nanja Smets, Tina Mioch, Mark Neerincx, Jurriaan van Diggelen, Francis Colas, **Ming Liu**, Francois Pomerleau, Roland


• Francis Colas, Srivatsa Mahesh, Francois Pomerleau, Ming Liu, Roland Siegwart, 3D Path Planning and Execution for Search and Rescue Ground Robots, in Proceedings of the IEEE International Conference on Intelligent Robots and Systems, IROS 2013, Best RoboCup Paper Award.


**Tech. Report**

• Towards Dynamic Object Detection using Key-point matching and Super-pixel Segmentation, 2010 May, ASL ETHZ.