ANNÉE 2016

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Bretagne Loire

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Traitement du Signal et Télécommunications
Ecole doctorale MATISSE
présentée par

Suman Raj BISTA

préparée à l'unité de recherche Inria Rennes - Bretagne Atlantique
Institut National de Recherche en Informatique et en Automatique

Indoor Navigation of Mobile Robots based on Visual Memory and Image-Based Visual Servoing

Thèse soutenue à Rennes
le 20 Décembre 2016
devant le jury composé de :

Youcef MEZOUAR
Professeur, SIGMA Clermont, Institut Pascal,
Clermont-Ferrand / Rapporteur

Cédric DEMONCEAUX
Professeur, Université Bourgogne Franche-Comté,
Le Creusot / Rapporteur

Anne SPALANZANI
Maître de Conférence HDR, Université
Grenoble Alpes / Examinateur

Eric MARCHAND
Professeur, Université de Rennes 1,
Rennes / Président du Jury et Examinateur

Paolo ROBUFFO GIORDANO
Directeur de Recherche, CNRS,
Rennes / Co-directeur de Thèse

François CHAUMETTE
Directeur de Recherche, Inria,
Rennes / Directeur de Thèse

YEAR 2016

THESIS / UNIVERSITY OF RENNES 1
under the seal of Bretagne Loire University

in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY

in Signal Processing and Telecommunication
of University of Rennes 1

Doctoral School MATISSE

Submitted by

Suman Raj BISTA

prepared at Lagadic Group, Inria Rennes - Bretagne Atlantique
French Institute for Research in Computer Science and Automation

Indoor Navigation of Mobile Robots based on Visual Memory and Image-Based Visual Servoing

Thesis Defended at Rennes in
20 December 2016

infront of the jury composed of :

Youcef MEZOUAR
Professor, SIGMA Clermont, Institute Pascal,
Clermont-Ferrand / Rapporteur

Cédric DEMONCEAUX
Professor, University of Burgundy Franche-Comté,
Le Creusot / Rapporteur

Anne SPALANZANI
Lecturer with Habilitation, University
Grenoble Alpes, Grenoble / Examinateur

Eric MARCHAND
Professeur, University of Rennes 1,
Rennes / President of Jury and Examinateur

Paolo ROBUFFO GIORDANO
Research Director, CNRS,
Rennes / Thesis Co-supervisor

François CHAUMETTE
Research Director, Inria,
Rennes / Thesis Supervisor

We have another chance to navigate, perhaps in a slightly different way than we did yesterday. We cannot go back. But we can learn.

Jeffrey R. Anderson,
The Nature of Things -
Navigating Everyday
Life with Grace.

Dedicated to My Parents.

Acknowledgment

First and foremost, I want to express my heartfelt gratitude, great respect, and sincere thanks to my supervisors, Dr. François Chaumette and Dr. Paolo Robuffo Giordano for their constant guidance and continuous support throughout this thesis. They give me the total autonomy to work during the entire thesis. The work would not have reached this level without their valuable inputs, encouragements and willingness to spare so much of their time. I also thank them for their detailed reviews, useful comments, and corrections when reviewing my papers, thesis manuscript, and slides that helped me to improve myself each time. I thank François sincerely for his timely advises as well as willingness to discuss and giving his time whenever I wanted. I am also grateful to Paolo for his suggestions, advises, and corrections in my scientific writing. It has been a real pleasure working with them.

I would also like to thank the jury members for their kindness of considering to participate in my Ph.D. defense committee and for putting forth their questions and concerns about the work. Many thanks to Prof. Youcef Mezouar and Prof. Cédric Demonceaux for earnestly reading the thesis and their valuable remarks, and Prof. Eric Marchand and Dr. Anne Spalanzani for investing their precious time on playing the role of the president of jury and the thesis examiner respectively.

This thesis has been supported by the Regional Council of Brittany (Conseil Régional de Bretagne) and OSEO Romeo 2 Project. I thank Inria for having provided me the excellent infrastructure and environment to pursue my doctoral studies.

I am grateful to Fabien Spindler for helping with familiarizing robotics platform and hardware, installing libraries and drivers, and solving software related issues. I thank him sincerely for his help and professionalism, for teaching me things that will be useful perhaps throughout my career. I also appreciate and thank Giovanni Claudio for his help in ROS, NaoQi and Pepper. My special thanks are due to Prof. Eric Marchand and Aurélien Yol for sharing me the codes of Mutual Information based Visual Servoing and Tracking respectively. I am also grateful to Souriya Trinh for his french translation of the synthesis of the thesis. I also thank Riccardo Spica for providing me the proper template of the manuscript.

I have received tremendous help and support from Lagadic team assistants Céline Gharsalli and Hélène de La Ruée throughout my Ph.D. It would like to thank them a lot. It would have been impossible to manage my way through the subtleties of the French bureaucracy and tedious administrative issues without their help. I thank Céline for being always-ready-to-help, for helping me to integrate into the team, and motivating to learn French. She was very friendly and always passing by our offices to enquire how we are doing. I also thank Hélène for her assistance and help in preparing missions and in smooth organization of activities related to the defense. I thank Sophie Viaud for her prompt response for administrative help. I also thank the staff at CMI, especially Emilie Gesnys for the clean and smooth progress of matters with the prefecture. I thank the SIC team for their fast support whenever there were some issues with the laptop.

I have had a great time in Lagadic Group. It was my pleasure to work in one of the biggest group in Inria with dynamic and friendly colleges. I am really grateful to them for the wonderful moments during my Ph.D. I must acknowledge François C, Paolo, Eric, Fabien, Marie, Alexandre, Vincent, Patrick, and Céline for sharing their experiences and knowledge. Thanks to Nicolas, Aurélien, Quentin, Rafiq, Ide Flore, Aly and Pierre for sharing the office with me in these three years. My special thanks to Aly, Le, Souriya, Mani and Nicolas for helping me on the issues related to University and CROUS. I am equally thankful to Aly, Vishnu, Souriya, Aurélien, Nicolas, Pierre, Giovanni, Pedro, Riccardo, Fabrizio, Ivan, Manikandan, François P, Lucas, Thomas, Joven, Antoine, Bertrand, Lesley, Jason, Quentin, Firas, Usman, Brayan, Noël, Ide, Nicolò, Julien, ... for all those never ending list of nice moments- chats, trips, parties, drinks, coffee breaks, lunches, sports and games, going out, movies ... I thank Bryan for the nice defense poster. Outside Lagadic, I would like to thank ADOC team with whom we organize events together. I am also thankful to Ashwin, Yogesh, Hrishikesh, Raghav, Arnab, Deepak, Nilay, and Cricket-Rennes for all those chats, drinks, encouragements, helps and Saturday cricket matches.

I am also grateful to all the people whom I meet and share nice moments during this thesis. My special thanks go to Sabina, Taman, Rupa, Bibek, Amit, Sandip, and Biraj for all those fun, happy and joyful moments, Nepalese meals, and outings we had together in Rennes, and for your encouragements and support. I am thankful to families of Merina and Keshav for inviting me during festival times. I am very grateful to all my other friends for the support and good moments they have given me during the past few years. To name a few: Danda Pani, Ajad, Bishesh, Achut, Lijia, Maya, Sharib, Deepak, Roshan, Daya Sagar, Qinglin, Francois R, Andru, Dadichi, Abinash, Shukra, Boney, Eachhya, Suman K, Umesh, Dipu, Gyanendra, Binod, Vesh, Prabhash, Subash, Ajay, ...

Finally and above all, my deepest gratitude goes to my family for their constant inspiration, support, and encouragement. It's because of your love, blessings, and selfless sacrifices, despite physically away from them, have rendered possible all my successes and achievements.

Synthèse en Français

La navigation est un problème fondamental en robotique mobile. Un robot devient autonome lorsqu'il est capable de se déplacer dans son environnement. Dans cette optique, les capteurs de vision sont cruciaux car ils permettent d'obtenir des informations détaillées sur l'environnement qui ne peuvent pas être obtenues par la combinaison d'autres types de capteurs. On observe par ailleurs un développement de l'usage des capteurs optiques en robotique grâce à la réduction des coûts de fabrication d'une part et d'autre part grâce à l'augmentation des capacités de calcul des ordinateurs ces dernières années. La réduction de la consommation électrique facilite également ces usages dans le domaine de l'embarqué. Couplés avec les rapides progrès obtenus dans le domaine de la vision par ordinateur, du contrôle et de la robotique, la navigation autonome a franchi un palier important. Bien que les progrès dans le domaine des systèmes de navigation basés vision sont significatifs, il reste cependant encore beaucoup de problèmes à résoudre avant d'obtenir concrètement des systèmes autonomes. Dans ce manuscrit, le chapitre 2 sera consacré aux théories fondamentales et aux diverses applications dans le domaine de la vision et de la robotique. L'état de l'art concernant la navigation en intérieur pour les robots mobiles sera présenté dans le chapitre 3.

Les approches classiques en navigation autonome sont basées sur une cartographie de l'environnement, la définition d'une trajectoire à réaliser, la localisation du robot et la minimisation de l'erreur entre la position courante et la trajectoire à suivre. Ce type de navigation basée vision repose sur la géométrie de l'environnement ainsi que sur d'autres informations dans l'espace métrique telles que l'estimation de la pose 3D par exemple. Les systèmes de navigation visuelle basés sur des approches différentes n'utilisent pas quant à eux une représentation explicite de l'environnement mais se consacrent plutôt à minimiser les erreurs de caractéristiques visuelles présentes dans l'image avec une relation directe avec la commande envoyée aux actionneurs. Cette approche basée image permet de réduire les temps de calcul, élimine la nécessité d'interpréter la structure de l'image et réduit également les erreurs induites par la modélisation de la caméra et des paramètres de calibration [CH06].

Par conséquent, l'approche que l'on souhaite développer dans cette thèse repose sur une représentation topologique de l'environnement, exprimée par un ensemble d'images clés que le robot devra successivement percevoir. Nous voulons démontrer qu'une cartographie et une localisation précise ne sont pas nécessaires pour la navigation visuelle. Des résultats utilisant ce paradigme ont été obtenus précédemment sur des robots mobiles terrestres évoluant dans des environnements urbains [ŠRDC09, DSRC11]. Nous souhaitons maintenant utiliser la même approche mais pour de la navigation en intérieur. Cette thèse cible donc en particulier une procédure de navigation basée seulement sur de l'information 2D dans l'image, ce qui permet une navigation lorsqu'aucune information 3D n'est disponible ou lorsqu'une reconstruction 3D n'est pas possible. Notre méthode peut ainsi être vu comme une alternative à la navigation basée SLAM.

L'objectif de cette thèse est donc de développer une approche de la navigation visuelle pour un robot mobile basée sur une mémoire visuelle. Comme l'on s'intéresse seulement à des robots mobiles destinés à un usage en intérieur, nos objectifs sont : a) de déterminer quelles caractéristiques visuelles sont appropriées pour la navigation en intérieur ; b) de développer une méthode complète regroupant la cartographie, la localisation et le contrôle du robot, applicable aux environnements en intérieur, prenant en compte les contraintes cinématiques ; et c) des expérimentations s'appuyant sur des scénarios intérieurs réels. Nous nous sommes essentiellement préoccupés de la navigation ciblée et dirigée, où le robot connaît son environnement pour naviguer à l'avance. La navigation sans cartographie de l'environnement au préalable est hors de la portée de cette thèse.

Aperçu Général de la Méthode Proposée

L'approche présentée ici reprend celle présentée dans [ŠRDC09], mais pour un environnement en intérieur avec différentes caractéristiques visuelles utilisant uniquement l'information 2D dans l'image, c'est-à-dire : - l'Information Mutuelle (MI) (Chapitre 5), les segments de droite (Chapitre 6) et la combinaison des caractéristiques points + droites (Chapitre 7). La méthode de navigation proposée est basée sur la mémorisation des images. Par conséquent, cet ensemble d'images doit être construit au préalable. Le processus général est divisé en 2 étapes : a) la cartographie et l'apprentissage effectués hors-ligne et b) la navigation en ligne.

Étape pour la Cartographie (Apprentissage)

Durant l'apprentissage, le robot est déplacé dans l'environnement de navigation sous la supervision d'un opérateur pour capturer les différentes images. De cette séquence d'apprentissage, un sous-ensemble d'images est prélevé, qui constitue ce que l'on appelle la mémoire d'images. Cette mémoire d'images se compose d'images clés/références représentant l'environnement considéré. Les images clés/références sélectionnées sont organisées dans un graphe d'adjacence, qui donne la représentation topologique de l'environnement. La première image de la séquence d'apprentissage est toujours choisie comme image clé/référence. Les autres images clés sont automatiquement sélectionnées de façon à être suffisamment différentes des autres images adjacentes. Cette mesure de différence est déterminée à partir d'un calcul de similitude, qui dépend de la caractéristique visuelle choisie. Dans le cas de caractéristiques géométriques locales, cette mesure de similarité est obtenue à partir de l'appariement et du suivi des caractéristiques visuelles. En revanche, il est nécessaire de recaler les parties communes dans l'image dans le cas des caractéristiques globales avant le calcul des mesures de similarité d'images. La dernière image de la séquence d'apprentissage est également stockée dans la base de données, ce qui permet de déterminer quand le robot doit s'arrêter à la fin de la navigation. Le schéma général de la phase de cartographie (apprentissage) est indiqué sur la Fig. 1. Les détails concernant la sélection d'images clés à l'aide de l'information mutuelle (MI) sont présentés dans la Sect. 5.2.2, à l'aide des segments de droite dans la section Sect. 6.2.2 et enfin à partir des caractéristiques points + droites dans la Sect. 7.2.1.

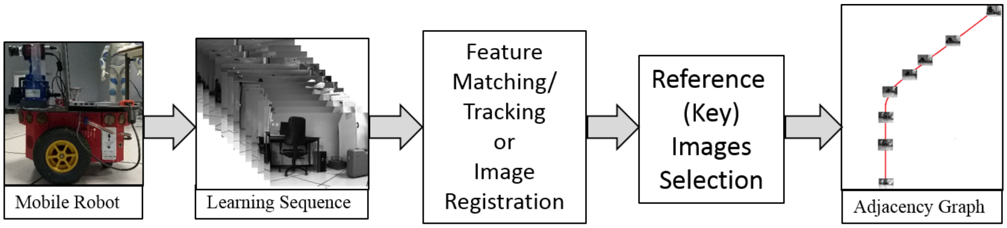

Mobile Robot
Learning Sequence
Feature Matching/ Tracking or Image Registration
Reference (Key) Images Selection
Adjacency Graph

Figure 1 – Étape pour la cartographie (apprentissage).

Étape de Navigation

Après la phase d'apprentissage, le robot est prêt à naviguer de façon autonome dans l'environnement cartographié. La tâche de navigation est alors divisée en deux sous-tâches : a) la localisation dans la carte des images et b) le contrôle du déplacement du robot. La vue d'ensemble du schéma général de la méthode de navigation proposée est représentée sur la Fig 2.

a) Localisation Qualitative dans la Carte des Images

Dans notre cas, la localisation dans la carte des images est qualitative plutôt que quantitative. Cela signifie que notre objectif est de trouver des images clés adjacentes qui correspondent le mieux avec la vue courante. Dans la Fig. 3, I_a représente la vue à la position courante du robot. La position de I_a dans la carte des images est située entre les images

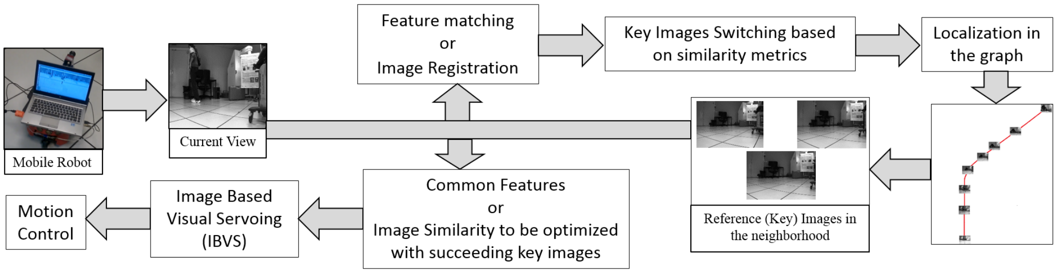


Figure 2 – Étape de navigation.

clés I_P et I_N. Les meilleures correspondances sont obtenues soit à partir d'appariement de caractéristiques locales, soit à partir du recalage des images. Pour de la navigation autonome, deux types de localisation doivent être réalisés : a) la localisation initiale (première localisation) b) les localisations suivantes. Les détails de la localisation qualitative dans la carte des images à l'aide de la méthode MI sont présentés dans la Sect. 5.2.3.1, à partir des segments de droite dans la Sect. 6.2.3.1 et à l'aide des caractéristiques points + droites dans la Sect. 7.2.2.1.

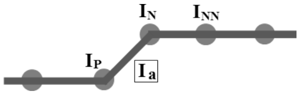


Figure 3 – Localisation dans la carte des images.

i) Localisation Initiale On appelle localisation initiale ou première localisation le processus de localisation du robot globalement dans la carte, où la vue actuelle est comparée avec l'ensemble des images situées dans la base de données en mémoire pour trouver les meilleures images clés adjacentes. La navigation commence par la localisation initiale, ce qui permet de démarrer le robot dans n'importe quelle position à l'intérieur de la carte des images. La localisation initiale peut également être appliquée pour récupérer le robot s'il est temporairement perdu. Si l'on définit $\mathfrak{f}(\ldots)$ comme la mesure de similarité, la position initiale de I_a peut être obtenue avec :

$$I_{k_i} = \arg\max_{I_k} \{\mathfrak{f}(I_a, I_{k_1}), \mathfrak{f}(I_a, I_{k_2}), \ldots\ldots, \mathfrak{f}(I_a, I_{k_n})\} \quad (1)$$

$$I_{k_j} = \arg\max_{I_k} \{\mathfrak{f}(I_a, I_{k_{i-1}}), \mathfrak{f}(I_a, I_{k_{i+1}})\} \quad (2)$$

$$\begin{aligned} I_P = I_{k_j} \text{ and } I_N = I_{k_i} \quad & \text{if } i > j \\ I_P = I_{k_i} \text{ and } I_N = I_{k_j} \quad & \text{if } i < j \end{aligned} \quad (3)$$

ii) Localisations suivantes Après la localisation initiale, les localisations suivantes peuvent être effectuées en comparant seulement quelques images clés adjacentes au voisinage de la position courante. Cette localisation va de pair avec le contrôle du déplacement du robot, ce qui permet de sélectionner les images clés appropriées pendant le contrôle de la navigation. Dans notre méthode, lors des localisations suivantes, l'image courante (I_a) est comparée avec les trois images clés : l'image clé précédente (I_P), l'image clé suivante (I_N) et la deuxième image clé suivante (I_{NN}). Le basculement des images clés se produit quand I_N précède I_a. Dans ce cas, I_N devient le nouveau I_P et I_{NN} devient le nouvel I_N. Si $\mathfrak{F}(...)$ désigne le critère de similarité à partir de trois-vues et $\mathfrak{f}(...)$ désigne le critère de similarité à partir de deux vues, le basculement des images clés est effectué lorsque :

$$\boxed{\mathfrak{F}(I_a, I_N, I_{NN}) > \mathfrak{F}(I_P, I_a, I_N) \text{ or } \mathfrak{f}(I_a, I_{NN}) > \mathfrak{f}(I_a, I_N). \qquad (4)}$$

Le critère à partir de deux vues est nécessaire pour éviter d'être perdu lorsqu'il n'est pas possible de calculer le critère à partir de trois vues.

b) Commande du Robot

Pour la tâche de navigation, le robot ne doit pas nécessairement atteindre avec précision chaque image clé de la trajectoire ou suivre avec précision la trajectoire apprise. Dans la pratique, le mouvement exact du robot doit être contrôlé par un module d'évitement d'obstacle [CC13]. Par conséquent, la vitesse en translation est maintenue constante et réduite à une valeur inférieure lors d'un virage. Ces virages sont automatiquement détectés en analysant la commande en vitesse en rotation. La vitesse en rotation est dérivée à l'aide des images clés et à l'aide de l'image courante à partir d'une loi de commande IBVS [CH06]. L'erreur à minimiser dans la méthode IBVS est obtenue à partir de la différence des positions des caractéristiques dans l'image ou à partir du gradient des caractéristiques globales. La navigation est qualitative parce que le robot suit approximativement la trajectoire apprise et ne converge pas vers toutes les images clés intermédiaires. Dans de nombreux cas au cours de la navigation, le basculement des images clés se produit avant convergence. Notre objectif est de faire en sorte que le robot suive approximativement le chemin appris, la méthode IBVS étant capable de garder l'erreur en-deçà d'une valeur désirée.

On définit un vecteur de caractéristiques visuelles courantes $\mathbf{s}$ et désiré $\mathbf{s}^\star$, la vitesse linéaire du robot est notée v_r, la vitesse en rotation notée ω_r (Fig. 4) et les jacobiennes associées $\mathbf{J}_v$ et $\mathbf{J}_\omega$ respectivement. Si l'on se réfère au calcul de dérivation présenté dans la Sect. 4, ω_r peut être calculé selon la loi de commande IBVS :

$$\omega_r = -\mathbf{J}_\omega^+(\lambda(\mathbf{s} - \mathbf{s}^\star) + \mathbf{J}_v v_r), \qquad (5)$$

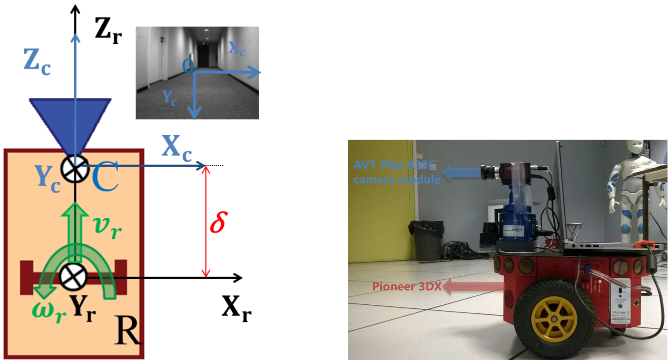


Figure 4 – Robot mobile vu de haut (orange) et équipé d'une caméra perspective (bleue) avec l'axe optique perpendiculaire à l'axe de rotation du robot (figure de gauche) ; photo de la plate-forme expérimentale (figure de droite).

où λ est un gain positif et $\mathbf{J}_{\omega}^{+}$ est la Pseudo-inverse de $\mathbf{J}_{\omega}$. Les calcul de $\mathbf{J}_{v}$ et $\mathbf{J}_{\omega}$ pour la méthode MI est présenté en Sect. 5.2.3.2, à partir des segments de droite dans la Sect. 6.2.3.2 et à partir des caractéristiques points+droites dans la Sect. 7.2.2.2.

Résultats et Discussions

Les contraintes dans notre méthode de navigation sont présentées dans la Sect. 4 et la présentation de la configuration utilisée lors des tests expérimentaux dans la Sect. 4. La cartographie a été réalisée hors ligne, alors que l'expérimentation de la méthode de navigation a été effectuée en ligne. v_r est maintenue constante et réduite au cours des virages, tandis que ω_r est contrôlé par (5.23) pour la méthode MI, (6.13) pour des segments de droite et (7.10) pour les caractéristiques points+droites. Les résultats expérimentaux sont présentés dans la Sect. 5.3 pour la méthode MI, Sect. 6.3 pour des segments de droite et Sect. 7.3 pour les caractéristiques points + droites.

Les résultats présentés montrent la viabilité de notre approche dans différents scénarios et suivant différentes contraintes. Le robot a été en mesure de suivre de manière autonome la trajectoire apprise à partir de la position de départ. La méthode IBVS a été en mesure de maintenir l'erreur sous une certaine limite. Le robot n'a pas exactement suivi le chemin appris en l'absence de l'utilisation des informations 3D et en l'absence de l'estimation du mouvement 3D. Ceci s'explique par le fait que l'on ne souhaite pas une navigation totalement précise, mais efficace et robuste. L'autre raison s'explique par l'approximation de la trajectoire par des lignes droites. Néanmoins, en se basant uniquement sur de l'information 2D, une navigation en intérieur est possible dans les couloirs et à l'intérieur de salles grâce à l'asservissement visuel qui permet de gérer de façon robuste ces différents types d'erreurs.

Conclusions et Travaux Futurs

Nous avons présenté une méthode complète de navigation et de cartographie qualitative en intérieur basée sur l'apparence. Notre méthode de navigation est exclusivement basée sur des mesures dans l'image 2D et sans s'appuyer sur une quelconque méthode de reconstruction 3D comme c'est le cas dans la plupart des publications existantes. Cela est possible grâce à une représentation topologique de l'environnement et à l'utilisation de la méthode IBVS pour le contrôle du robot. L'utilisation simultanée de la vision et du contrôle du robot en boucle fermée via la méthode IBVS permet au robot de suivre la trajectoire apprise tout en gardant une erreur en deçà d'une certaine limite définie et cela même en l'absence d'informations 3D.

Les résultats de cette thèse, bien qu'encourageants, montrent également un certain nombre de limitations qui affectent notre approche. La première est liée à la détection de caractéristiques visuelles et à leur appariement. Ce problème peut être atténué en combinant plusieurs types de caractéristiques visuelles. La deuxième est liée à l'évitement d'obstacles. Cependant, nous pouvons facilement intégrer dans notre méthode un module d'évitement d'obstacles utilisant des lasers comme cela est proposé par [CC13]. La dernière limitation est liée à des incertitudes dans la navigation lorsque la scène est faiblement texturée ou lorsqu'elle présente des textures répétitives. Ce problème peut être résolu en utilisant la cinématique du robot et l'utilisation de filtres probabilistes pour filtrer la vitesse en rotation.

Les détails des conclusions et des perspectives sont discutés au Chapitre 8.

Contents

*

Notations

General notation conventions

Throughout this thesis, the following notation conventions will be used:

- Scalar quantities are represented by lowercase symbols such as a, b and so on.
- Vector quantities and matrices are represented by bold symbols such as $\boldsymbol{a}$, $\boldsymbol{b}$ and so on.
- The notation $(a,\ b)$ or $\begin{bmatrix} a \\ b \end{bmatrix}$ indicates a vertical concatenation of elements (scalars, vectors or matrices) and $[a\,b]$ or $[a\,|b]$ for horizontal concatenations.
- $\boldsymbol{I}_n$ denotes the identity matrix of order $n \times n$.
- $[\boldsymbol{t}]_\times$ denotes skew-symmetric matrix of 3D vector t.
- $[\boldsymbol{a}]_\times$b denotes cross product $a \times b$.
- $det(\boldsymbol{M})$ denotes determinant of $\boldsymbol{M}$.
- $\boldsymbol{M}^{-1}$ denotes inverse of matrix $\boldsymbol{M}$ if $\boldsymbol{M}$ is a square matrix and $det(\boldsymbol{M})\neq 0$.
- $\boldsymbol{M}^+$ is pseudo-inverse of $\boldsymbol{M}$ i.e. $\boldsymbol{M^+} = (\boldsymbol{M}^T\boldsymbol{M})^{-1}\boldsymbol{M}^T$ if $\boldsymbol{M}$ has all linearly independent columns $\mathbf{M}$.
- $\boldsymbol{M}^T$ denotes transpose of matrix $\boldsymbol{M}$.
- If $\boldsymbol{w}$ denotes some matrix/vector in first view, then $\boldsymbol{w'}$ and **w”** denotes the same but for second and third view respectively.
- $(m.n)$ represents equation number (m is chapter number, n is serial number of equation in the chapter m).

Acronyms and abbreviations

2D	2 Dimensional.
3D	3 Dimensional.
arg	argument.
BRISK	Binary Invariant Scalable Keypoints.
CC	Cross-Correlation.
DOF	Degree of Freedom.
ED	Edge Drawing.
EKF	Extended Kalman Filter.
FAST	Features from Accelerated Segment Test, a corner detector.
Fig.	Figure.
FOV	Field of View.
IBVS	Image-Based Visual Servoing.
ICRA	International Conference in Robotics and Automation.
LBD	Line Band Descriptor.
LSD	Line Segment Detector.
MI	Mutual Information.
KLT	Kanade-Lucas-Tomasi feature tracker.
max	Maximize.
PBVS	Pose-Based Visual Servoing.
RANSAC	RANdom Sample Consensus.
Ref./ref.	Refer or Reference.
Sect.	Section.
SfM	Structure from Motion.
SLAM	Simultaneous Localization and Mapping.

SSD	Sum of Squared Difference.
SIFT	Scale Invariant Feature Transformation.
SURF	Speeded-Up Robust Features.
SVD	Singular Value Decomposition.
V-SLAM	Vision based SLAM.
VO	Visual Odometry.
VS	Visual Servoing.
w.r.t	with respect to.
ZNCC	Zero Mean-normalized Cross-Correlation.

CHAPTER 1

Introduction

1.1 Background

Navigation is a fundamental problem in mobile robotics. For the robot to be autonomous, it should be able to navigate in the environments on its own. Vision is one of the most powerful and important sensor in this aspect as it provides the detailed information about the environment which may not be available using combinations of other types of sensors. Vision sensors are developing and becoming cheaper in one hand, and processing power of computers have increased significantly in the recent years and that too with lesser power consumption in another hand. Together with them, the rapid progress and tremendous research in the field of computer vision, control, and robotics, make it possible to have autonomy to a greater extent. The achievements in the area of vision-based navigation systems are significant; however, there is still a long way to go, and this is still an open problem.

1.2 Motivation

Classical approaches for navigation are based on a mapping of the environment, the specification of a trajectory to follow, the localization of the robot, and the regulation of the error between this localization and the specified trajectory. Such type of the visual navigation system relies on the geometry of the environment and on other information in metric space such as 3D pose estimation. Alternative visual navigation systems use no explicit representation of the environment, where an error signal measured directly in the image is mapped to actuator commands. Using image-based visual servoing approach reduces computational delay, eliminates the necessity for image interpretation, and errors due to camera modeling and calibration [CH06]. Therefore, the approach we want to develop in the thesis is based on a topological representation of the environment, expressed as a set of key images that the robot will have to perceive successively. We want to demonstrate that accurate mapping and localization are not necessary for visual navigation. Previous results have been obtained using this paradigm on ground mobile robot evolving in urban environments [ŠRDC09, DSRC11]. Now, we want to use the same approach for the indoor environment. Hence, this thesis aims in a navigation process based on 2D image information without using any 3D information, which particularly makes navigation possible when there is no 3D information available or when 3D reconstruction is not possible. Therefore, our method can be seen as the alternative to SLAM-based navigation.

1.3 Objectives and Goals/ Scope of the Thesis

The goal of the thesis is to develop a visual navigation approach for a mobile robot based on visual memory. As for an indoor mobile robot, our objective is

- to determine what are the good visual features for navigating indoors.
- to develop complete methods for mapping, localization, and control, applicable to indoor environments, taking kinematics constraints into account.
- to experiment in real indoor scenarios.

We are basically concerned with goal-directed navigation where the robot has knowledge of the environment to navigate in advance. The navigation outside the mapped environment is out of the scope of this thesis.

1.4 Structure of the Thesis

This thesis is structured into three parts: The first part deals with the introduction and review of some basic theory, practices, and literature concerned with the mobile robot navigation. The second part consists of our contribution. The final part deals with conclusions and future perspectives. In the following, we propose a brief summary of the content of each part.

Outline of Part I

Part I deals with the introduction, basic fundamental theories and the state of the art for indoor mobile robot navigation.

Chapter 2 deals with some preliminaries related to Computer Vision and Robotics and how they can be incorporated together by Visual Servoing. We start by summarizing the basic model of image formation, camera calibration, features detection and correspondences, the geometrical relationships between images of the same scene taken from different camera points of view, Structure from motion, Visual SLAM and Visual Odometry. The chapter continues by presenting the standard techniques used for modeling the kinematics of a robot and for controlling its motion. Finally, we focus on different Visual Servoing schemes that are applicable for vision-based control.

Chapter 3 deals with some state of the art techniques, paradigms, and works for visual navigation of mobile robots. We first review in different aspects like operating environment (indoor or outdoor), knowledge of environment (map and mapless), model of the environment (appearance-based or model-based), and different mapping paradigms (metric maps and topological maps). Finally, we present and discuss some state of the art methods on appearance-based navigation for indoor mobile robots on the basis of different types of features used.

Outline of Part II

The second part of this thesis focuses on our contributions. The content in this part are the subjects of our publications [BRGC16a, BRGC16b] and article under review [BRGCew].

In Chapter 4, we give an overview of the proposed framework for mapping, localization and control, our constraints and our experimental setup.

In Chapter 5, Mutual information-based navigation is presented, extending the work of [DM13] with automatic key images selection, initial localization in the map, switching of key images and use of multiple key images for control.

In Chapter 6, we present feature-based navigation using 2D line segments. The entire navigation framework depends upon 2D matching of general line segments.

In Chapter 7, we extend the work of the previous chapter by combining line segments with feature points, where we show that combination increases the robustness of navigation.

Outline of Part III

The final part of this thesis deals with the conclusions and references.

In Chapter 8, we give conclusions of our contributions and discuss future perspectives and open issues.

Part I

Preliminaries and State of Art

CHAPTER 2

Fundamentals of Computer Vision and Robotics

THIS chapter presents some essential theory and concepts related to computer vision and robotics necessary for visual navigation of mobile robots. We begin this chapter with basics of computer vision. First, we talk about image formation, camera parameters, and camera calibration in Sect. 2.1.1. Then we discuss some popular features extraction and matching techniques in Sect. 2.1.2, which is indeed basis of our work. Next, we present two view and three view projective geometry in Sect. 2.1.3, which is used in our work for the verification of feature correspondences. After that, we talk basics of estimating camera motion and 3D reconstruction of scene in Sect. 2.1.4, Visual SLAM in Sect. 2.1.5 and Visual Odometry in Sect. 2.1.6. After computer vision, we move our focus to mobile robots modeling and motion control applicable for non-holonomic wheeled robots navigation in Sect. 2.2. Finally, we conclude this chapter with visual servoing framework that puts together computer vision and robotics in Sect. 2.3. In particular, we concentrate on Image Based Visual Servoing (IBVS) techniques, which is the backbone of the motion control in our work.

2.1 Computer Vision

2.1.1 Cameras and Image Formation

2.1.1.1 Vision Sensor (Camera) Model

A camera model is a function which maps a 3-dimensional world onto a 2-dimensional plane, called the image plane. Generally, this function is designed to model the real world physical camera that should take perspective into account. Perspective is the property that objects far away from us appear smaller than objects closer to us, which is the case with human vision, and with most photometric cameras in the real world. Photometric cameras using an optical lens can be modeled as a pin-hole camera. A pinhole camera is a light-tight box or chamber with a black interior and a tiny hole in the center of one end which creates an image of the outside space on the opposite end of the box by the principle of rectilinear propagation of light. The most common model for the general camera is the pin-hole camera with the perspective projection [HZ03, Sze10] as shown in the Fig. 2.1.

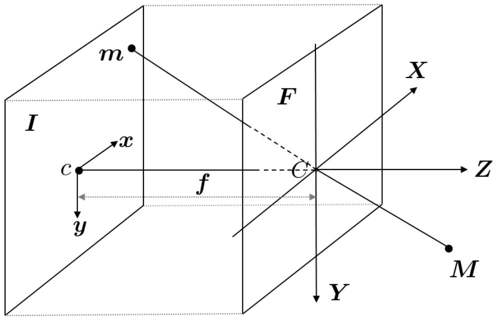


Figure 2.1 – Pin-hole camera with perspective projection.

If (X,Y,Z) represents a three-dimensional camera coordinate system, and (x,y) represents image plane, this relationship for the projection of 3D feature of the object onto image plane is given by

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \tag{2.1}$$

where f is the focal length of the lens and $\lambda = Z$ is a homogeneous scaling factor. The range information is lost in this projection, but the angle or orientation of the object point can be obtained if the focal length is known and the lens does not cause distortion/or distortion is corrected.

2.1.1.2 Perspective Camera Parameters and Calibration

a) Intrinsic Parameters [HZ03, Sze10] Most of the current imaging systems define the origin of the pixel coordinate system at the top-left pixel of the image as shown in Figs. 2.2-2.3. However, it was previously assumed in (2.1) that the origin of the pixel coordinate system corresponds to the principal point (c_x, c_y), located at the center of the image. A conversion of coordinate systems is thus necessary. Also, in (2.1), it was implicitly assumed that the pixels of the image sensor are square and pixels are not skewed. However, both assumptions may not always be valid. Hence, these imperfections of the imaging system (Fig. 2.3) should be taken into account in the camera model. If s_x and s_y represent the effective size of pixels in horizontal and vertical directions respectively and γ represents the skew, the projection mapping can be now written as:

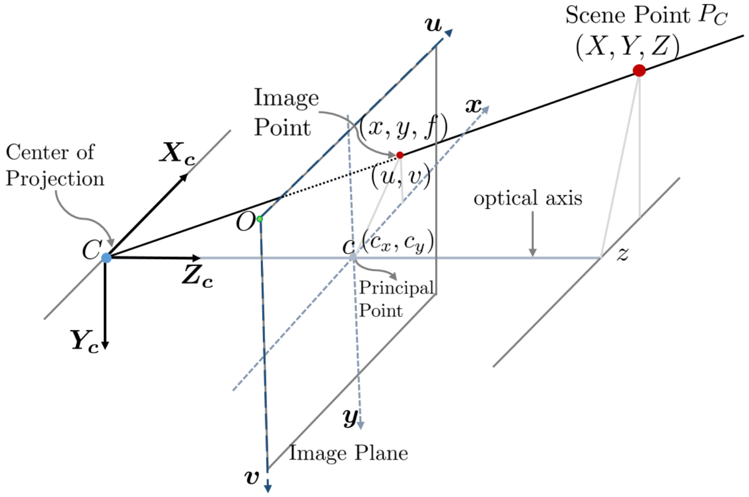


Figure 2.2 – Image formation with perspective projection. The 3D coordinate system has origin at C (Center of Projection of camera).

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \tag{2.2}$$

where $f_x = \frac{f}{s_x}$ and $f_y = \frac{f}{s_y}$ represents the effective focal length along height and width of the image. For ideal pinhole camera $f_x = f_y$. In most of modern days cameras, skew can be safely assumed to be zero. These parameters f_x, f_y, γ, c_x, and c_y are known as *Intrinsic Parameters*, which are internal and fixed to a particular camera/digitization setup and allow a mapping between camera coordinates and pixel coordinates in the image frame. These parameters represent focal length, image sensor format, and principal point. These parameters can be represented in a 3×3 intrinsic parameter matrix $\mathbf{K}$ as

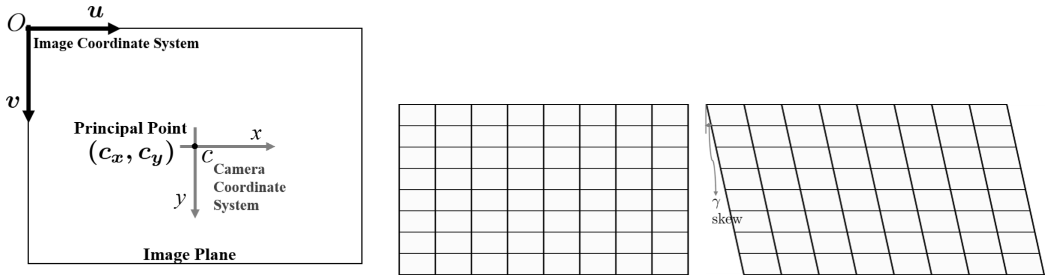


Figure 2.3 – Camera and Image coordinate systems (Left), Non Square Pixels (Middle), and Skew of Pixels. These factors also should be taken into account while modeling the image formation by the camera.

$$\mathbf{K} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.3}$$

These intrinsic parameters can be easily obtained from camera calibration.

b) Distortion Parameters [WCH+92, MK04] Up to this point, we assumed that a point in 3D space, its corresponding point in image and the camera's optical center are collinear. This linear projective equation is sometimes not sufficient, especially for low-end cameras (such as web-cams) or wide-angle cameras; lens distortion has to be considered. They are mainly caused either due to imperfect lens shape or improper lens assembly. *Radial distortion* (Fig 2.4b,c) is symmetric in which caused an inward (also known as *barrel distortion*, Fig 2.4b) or outward (also known as *pincushion distortion*, (Fig 2.4c) displacement in radial direction of a given image point from its ideal location (Fig. (Fig 2.4a). This type of distortion is mainly caused due to faulty radial curvature of lens. *Decentering distortion* (Fig 2.4d) is non-symmetric in which the ideal image points are distorted in both radial and tangential directions. This type of distortion is caused due to the misalignment of the lens elements when the camera is assembled, where optical centers of lens elements are not strictly collinear. Such distortion can be corrected by following:

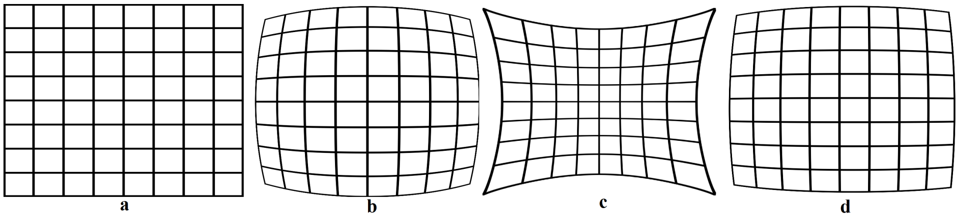


Figure 2.4 – a. Ideal Image Points, b. Barrel Distortion c. Pincushion Distortion, and d. Decentering Distortion.

$$\begin{aligned} x' &= x + x(k_1r^2 + k_2r^4 + k_3r^6 + \ldots) + [2p_1y + p_2(r^2 + 2x^2)](1 + p_3r^2 + \ldots), \\ y' &= y + y(k_1r^2 + k_2r^4 + k_3r^6 + \ldots) + [2p_2x + p_1(r^2 + 2y^2)](1 + p_3r^2 + \ldots), \end{aligned} \tag{2.4}$$

where (x', y') is the corrected position of input image at (x, y), $r = \sqrt{x^2 + y^2}$ is the radial distance, $k_i's$ are the coefficients of radial distortion, and $p_j's$ are the coefficients of the tangential distortion. These distortion parameters can be easily obtained from the camera calibration. In practice, first few coefficients k_1, k_2, k_3, p_1, and, p_2 are used for the distortion correction.

c) Extrinsic Parameters [HZ03, Sze10] If $\boldsymbol{p}$ is 2×1 vector representing 2D projected image pixel and $\boldsymbol{P}_c$ is 3×1 vector that represents the corresponding 3D point in the camera coordinate system, (2.2) can be written as

$$\lambda \begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix} = \boldsymbol{K}\boldsymbol{P}_c, \tag{2.5}$$

where it is assumed that the world coordinate system and the camera coordinate system are same. However, in practice it may not be always true as it is preferred to use the world coordinate system to locate 3D points. Some transformation between world coordinate frame and camera coordinate frame is necessary. This can be done by introducing two extra parameter sets, rotation and translation as shown in Fig. 2.5.

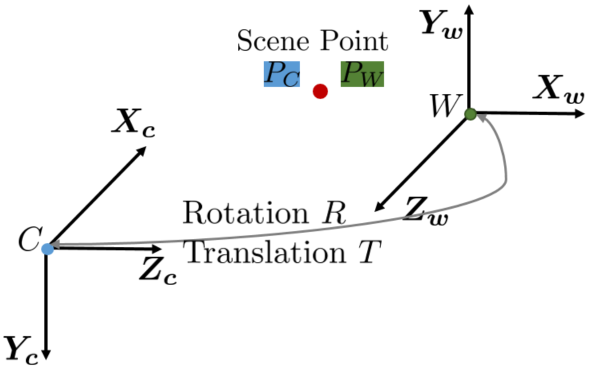


Figure 2.5 – World coordinate system and Camera coordinate system.

If $\boldsymbol{P}_W$ is 3×1 vector representing 3D points in world coordinate system, $\boldsymbol{R}$ is a 3×3 rotational matrix and $\boldsymbol{T}$ is a 3×1 translational vector that represents the position of the origin of the world coordinate system expressed in coordinates of the camera coordinate system, the relationship between $\boldsymbol{P}_c$ and $\boldsymbol{P}_W$ can be written as

$$\boldsymbol{P}_c = \boldsymbol{R}\boldsymbol{P}_W + \boldsymbol{T}. \tag{2.6}$$

From (2.27) and (2.6), we have

$$\lambda \begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix} = \boldsymbol{K}(\boldsymbol{R}\boldsymbol{P}_W + \boldsymbol{T}). \tag{2.7}$$

This equation can be rewritten as

$$\lambda \begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix} = \boldsymbol{K}[\boldsymbol{R}|\boldsymbol{T}] \begin{bmatrix} \boldsymbol{P}_W \\ 1 \end{bmatrix} = \boldsymbol{P} \begin{bmatrix} \boldsymbol{P}_W \\ 1 \end{bmatrix}, \tag{2.8}$$

where $\boldsymbol{P}$ is 3×4 perspective projection matrix, which is also known as camera matrix that is used to transform the point in a world coordinate to the point in the image . The position of $\boldsymbol{C}$ expressed in world coordinates is obtained from

$$\boldsymbol{C} = -\boldsymbol{R}^{-1}\boldsymbol{T} = -\boldsymbol{R}^T\boldsymbol{T}. \tag{2.9}$$

Therefore, R and T can sufficiently define the location and orientation of the camera with respect to the world frame. These three 3D rotational angles and a 3D translation vector are also known as *Extrinsic Parameters*, which are external to the camera and may change with respect to the world frame.

From 2.8-2.9, we have

$$\boldsymbol{P} = \boldsymbol{K}\left[\boldsymbol{R}\,|\,\boldsymbol{T}\right] = \boldsymbol{K}\left[\boldsymbol{R}\,|\,-\boldsymbol{R}^T\boldsymbol{C}\right]. \tag{2.10}$$

If this matrix $\boldsymbol{P}$ is known, one can easily back project 2D image point into 3D. For 2D image point $\boldsymbol{p}$, there exists a collection of 3D points that are mapped and projected onto the same point $\boldsymbol{p}$. This collection of 3D points constitutes a ray connecting the camera center $\boldsymbol{C}$ and $\boldsymbol{p}$. With two or more cameras, the unique 3D position can be estimated, which will be described in later sections. Hence, the accurate estimation of $\boldsymbol{P}$ is essential for 3D estimation. For this accurate camera calibration is necessary.

d) Camera Calibration Primarily, camera calibration is finding the quantities internal to the camera that affect the imaging process. These quantities are intrinsic parameters and distortion parameters. Various camera calibration techniques exist in the literature [HZ03, Sze10, MK04]. The most simpler, flexible, accurate and widely used is the one proposed by [Zha00]. Various toolboxes for this method exit in ViSP [MSC05], OpenCV [Ope] and Matlab [Bou] that uses known pattern like black-white chessboard, symmetrical and asymmetrical circle pattern. The overall process of calibration can be summarized into following steps: [MK04]

1. Print a pattern and attach it to a planar surface;
2. Take a few images of the model plane under different orientations by moving either the plane or the camera;
3. Detect the feature points in the images;
4. Estimate the intrinsic parameters and all the extrinsic parameters;
5. Refine all parameters, including lens distortion parameters using non-linear iterative estimation methods.

In all of our work, we will consider that camera is calibrated.

2.1.2 Features and Correspondences

Feature detection, description and matching are essential components of many computer vision applications [Sze10]. Several feature detectors and descriptors have been proposed in the literature with a variety of definitions for what kind of regions in an image is potentially interesting (i.e., a distinctive attribute). On the basis of them, various matching methods are available to find the corresponding interest regions.

2.1.2.1 Global and Local Features

In computer vision tasks, we need to represent the image by features extracted from them. There are generally two methods to represent images: global features and local features [HAA16]. In the global feature representation, the image is represented by a single multidimensional feature vector using the information from all pixels of images such as intensity, color, texture, histograms, gradients, some types of transformation in image like integral image and Fourier descriptors, specific descriptor extracted from some filters applied to the image like GIST descriptor [TMFR03] and so on. These single vectors, one from each image can be used to compare the two images. Sometimes, one extra vector using combined features from both images like joint histograms are also used in addition to the feature vectors from single image. Global features are compact, much faster to compute and require small amount of memory. However, they are not invariant to significant transformations and sensitive to clutter and occlusions. On the other hand, in the local feature representation, the image is represented by more than one multidimensional feature vectors (also known as local feature descriptors) based on their local structures and textures from a set of salient regions in the images like keypoints, corners, keylines while remaining invariant to some sort of viewpoint and illumination changes. Local features have following characteristics: [Sze10]

1. They are robust to occlusion and cluttering as they are local.
2. They can differentiate a large database of objects efficiently because local structures are more distinctive and stable than other structures in smooth regions.
3. They may be in hundreds or thousands in a single image.
4. Real-time performance can be achievable.
5. They exploit different types of features in different situations.

However, they usually require a significant amount of memory because the image may have hundreds of local features. Nevertheless, the method exits to aggregate these descriptors into compact representation and reducing dimension of the feature descriptors.

2.1.2.2 Feature Detection

The feature extraction/detection process should be repeatable, accurate, and distinctive so that the same features are extracted on two images showing the same object and different image structures can be told apart from each other. Besides that, it is desired to have features robust to the geometric transformations (translation, scaling, rotation and shifting), photometric changes (brightness and exposure), compression artifacts, and noise.

a) Points and blob detectors

Point features can be used to find a sparse set of corresponding locations in different images. Various types of point features have been proposed in the literature to detect salient points in the images, also known as keypoints. Such keypoints permit matching even in the presence of occlusion, and large scale and orientation changes. In other hand, edges and general curves are suitable for describing the contours of natural objects and straight lines for the man-made world.

Corner detectors like Moravec's Detector [Mor77], Harris Detector [HS88], SUSAN (Smallest Univalue Segment Assimilating Nucleus) Detector [SB97], and FAST (Features from Accelerated Segment Test) Detector [RPD10], and blob detector like Hessian Detector [MTS$^+$05] are single-scale point detectors, which perform poorly when there is change of scale.

Moravec's Detector [Mor77] is one of the earliest corner detector that uses sum of squared differences (SSD) between the centered patch and the neighboring image patches to determine the point low self-similarity, which he defined as corners. The main disadvantage is that detector is not isotropic that leads to detect some edges as corners and not rotational invariance.

Harris Corner Detector [HS88] improves Moravec's corner detector by considering the differential of the corner score with respect to direction directly, instead of using shifted patches. Since corners are the intersection of two edges, it represents a point in which the directions of these two edges change. Hence, the gradient of the image (in both directions) have a high variation as shown in Fig. 2.6(left), which has been exploited in Harris Corner detection method.

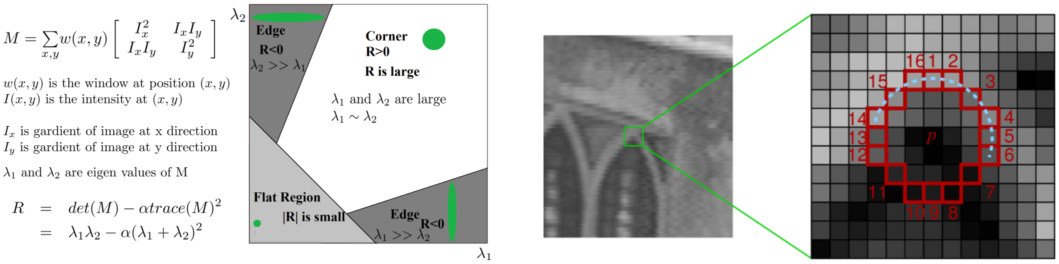

$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}$
$w(x,y)$ is the window at position (x,y)
$I(x,y)$ is the intensity at (x,y)
I_x is gardient of image at x direction
I_y is gardient of image at y direction
λ_1 and λ_2 are eigen values of M
$R = det(M) - \alpha trace(M)^2 = \lambda_1\lambda_2 - \alpha(\lambda_1+\lambda_2)^2$
λ_2
Edge
R<0
$\lambda_2 >> \lambda_1$
Corner
R>0
R is large
λ_1 and λ_2 are large
$\lambda_1 \sim \lambda_2$
Flat Region
|R| is small
Edge
R<0
$\lambda_1 >> \lambda_2$
λ_1
16 1 2
15
3
14
4
13
5
12
6
p
11
7
10 9 8

Figure 2.6 – Harris Corner Detection [HS88] (left) and FAST Corner Detection [RPD10] (right).

[SB97] introduced a generic low-level image processing technique called **SUSAN** instead of using image derivatives to compute corners. The center pixel is referred to as the nucleus. Pixels having almost the same brightness as the nucleus are grouped together and the resulting area is termed USAN (Univalue Segment Assimilating Nucleus). This method is fast, and invariant to translation and rotation. But, it requires fixed global threshold making it unsuitable for general applications.

Hessian Detector [MTS+05] is based upon second order derivatives that have good performance in computation time and accuracy. However, using second order derivatives in the detector is sensitive to noise.

FAST corner detector [RPD10] uses a circle of 16 pixels to classify whether a candidate point $\boldsymbol{p}$ is actually a corner. Each pixel in the circle is labeled from integer number 1 to 16 clockwise as shown in Fig. 2.6(right). If a set of n contiguous pixels in the circle are all brighter than the intensity of candidate pixel plus a threshold value or all darker than the intensity of candidate pixel minus threshold value, then $\boldsymbol{p}$ is classified as corner. The main advantage of this corner detector is its computational efficiency in terms of time and resources. Moreover, when machine learning techniques are applied, superior performance can be realized, which makes it faster than many other well-known feature extraction methods. However, it is not invariant to scale changes and not robust to noise.

In all of above methods, the problem of detecting multiple interest points in adjacent locations is overcame by using non-maximal suppression. Using Gaussian scale pyramids, scale invariance can be achieved.

In **Laplacian of Gaussian (LoG)** [Lin98], the scale space representation of the image is obtained by convolving the image by a variable scale Gaussian kernel. The interest point is obtained by searching for 3D (location + scale) extrema of the LoG function. Also, LoG is circular symmetric, which means it is naturally rotational invariant. However, the computation of LoG is expensive.

To overcome the expensive computation problem of LoG, Lowe [Low04] approximate LoG with **Difference of Gaussian (DoG)** for **Scale-Invariant Feature Transform (SIFT)** algorithm.

Harris-Laplace [MS04] detector combines Harris corner detector and a Gaussian scale space representation to detect scale-invariant corners. The points are highly repeatable that are invariant to scale changes, rotation, illumination, and addition of noise. However, the Harris-Laplace detector returns a much smaller number of points compared to the LoG or DoG detectors and it fails in the case of affine transformations.

Hessian-Laplace is similar to Harris-Laplace, but it uses Hessian blob-detector instead of Harris corner. In comparison to Harris-Laplace, this detector extracts large number of features that cover the whole image at a slightly lower repeatability. In **Speeded-Up Robust Features (SURF)** [BETVG08], Bayes uses Fast Hessian detector by approximating LoG by box filters and the Hessian by integral images.

In **Oriented FAST and Rotated BRIEF (ORB)** [RRKB11] feature detector, FAST-9 (circular radius of 9) with a scale pyramid of the image has been used. FAST features at each level in the pyramid are filtered by using Harris corner at each level. Finally, using the moment of image patch, the orientation of the interest point is calculated to obtain Oriented FAST. **Binary Robust Invariant Scalable Keypoints (BRISK)** [LCS11] also use Gaussian scale pyramid with FAST features filtered with FAST score to generate scale invariant keypoints.

In [YH14] **Gabor wavelets** are used to generate high accuracy interest points that can adapt to various geometric transformations.

The keypoint extracted above assumes that localization and scale is not affected by affine transformation of image structures. They can handle the problem of affine invariance up-to some extent only because most of the methods fail in presence of significant affine transformation. Various affine invariant detectors have been proposed on the basis of affine normalization or affine shape adaptation to the methods described above. Several state-of-the-art affine detectors [MS04] have been analyzed in [MTS$^+$05]. Besides, there are some region detectors like **Maximally Stable Extremal Regions (MSER)** proposed by [MCUP04], which is affine invariant. MSER is a blob detector, which extracts a number of co-variant regions from an image based on connected component of some gray-level sets of the image, known as extremal regions. Such extremal regions detect structures in multi-scale, and are affine invariant and it doesn't matter if the image is warped or skewed. However, they are sensitive to natural lighting effects as change of day light or moving shadows.

As a compromise between speed and performance regarding to tracking and matching, we will use FAST Corners and SURF points in Chapter 7.

b) Line detectors

Another important local feature especially in man-made environment are line segments. Line segments in the image can be detected classically from edge maps, typically by Canny Edge detector by using Hough Transformation. **Hough Transformation** [DH72] is based upon voting and global search to obtain line segments. However, this method is slow, sensitive to parameters like sampling step, thresholding, and detects many false lines. There are many variants of Hough transform like Randomized Hough transform [XOK09], Probabilistic Hough transform [KEB91, MGK00], Elliptical Gaussian kernel-based Hough transform [FO08] and [LLL94], [KDCA07] etc., each trying to remedy different shortcomings of the standard Hough transform. [CY96] extract line segments from the edge map by detecting chain of pixels followed by walk over the chain. However, these methods still require manual parameter tuning for generating edge maps and also detect many false positives.

[BHR86, BGL] use the pixels' gradient orientations rather than edge maps. Although, this idea laid the foundation of some of recent line detectors, these methods detect too many false positive. [DMM07] proposed a parameterless line detector that controls the number of false positives on the base of this idea. To remove the false positive, they count the number of aligned pixels in a certain orientation and accept the set of pixels as a line segment if the observed structure is perceptually meaningful. This line validation method is known as Helmholtz principle [DMM07]. Although this method does not detect false positive, it is too slow and it does not detect many valid lines which are short.

By extending work of [BHR86] for line segment generation and combining it with line validation method using Helmholtz principle [DMM07], [GJMR10] proposed a parameterless line detection algorithm, called the **Line Segment Detector (LSD)**, that produces accurate line segments, and also controls the number of false detections. Although LSD produces good results for most types of images in real time, it fails especially in images where the background contains some white noise.

In [AT11], authors propose a fast, parameterless line segment detector known as **Edge Drawing (ED) Lines**, which produces robust and accurate results, and runs around 10 times faster than LSD. In this method, line segments are detected in three steps: a) Using Edge drawing algorithm [TA12] to produce a set of clean, contiguous chains of pixels, known as edge segments, b) Extraction of line segments from the edge segments using Least Squares Line Fitting, and (c) validation of lines using Helmholtz principle [TA12] to eliminate false line segment detections. We will use ED line detector to detect lines for our work as explained in Chapters 6 and 7.

2.1.2.3 Feature Description

Once a set of interest regions has been extracted from an image, their location, scale, orientation, and their content information need to be encoded in a descriptor that is suitable for discriminative matching. There are various descriptors in the literature for defining interest points and lines. Here, we will discuss only few that have been used widely.

a) Keypoint Descriptors

SIFT descriptor [Low04] is a vector of histograms of image gradients. The region around the interest point is divided into a 4×4 square grid at the appropriate scale and orientation. Each cell yields a histogram with 8 orientation bins that are arranged in 128-dimensional vector. This vector is then normalized to unit length in order to enhance invariance to affine changes in illumination. A threshold of 0.2 is applied so as to reduce the effects of non-linear illumination and the vector is again normalized. SIFT feature descriptor is invariant to translation, rotation, uniform scaling, orientation, and partially invariant to affine distortion and illumination changes.

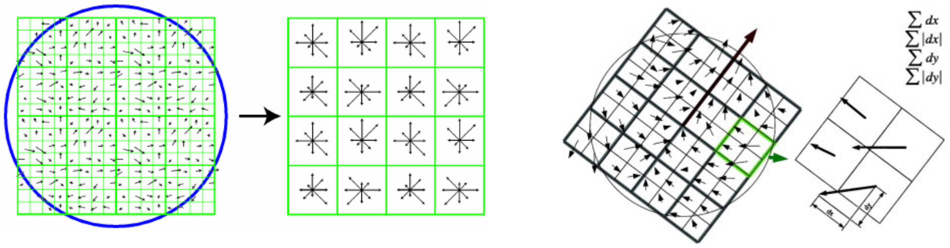


Figure 2.7 – SIFT descriptor [Low04](left) and SURF descriptor [BETVG08](right).

Gradient Location-Orientation Histogram (GLOH) [MTS+05] is similar to SIFT, but uses log-polar grid instead of Cartesian grid. It uses 16 orientation bins for each of 17 spatial bins. The spatial bins are of radius 6, 11, and 15. Except the central bin, other two bins are further divided into eight angular bins. The 272-dimensional histogram is then projected onto a 128-dimensional descriptor using Principal Component Analysis (PCA). Based on the experimental evaluation conducted in [MTS+05] , GLOH outperforms SIFT descriptors by small margin.

SURF descriptor [BETVG08] uses Haar wavelet. A neighborhood of size $20s \times 20s$ is taken around the key-point, where s is the scale. It is then divided into 4×4 subregions. For each subregion, horizontal and vertical wavelet responses are taken and a 4 element vector is formed. Hence, the SURF descriptor is a 64-dimensional vector. The main advantage of the SURF descriptor compared to SIFT is the processing speed keeping its performance comparable to SIFT. SURF is good at handling images with blurring and rotation, but not good at handling viewpoint change and illumination change. To improve the distinctiveness, it can be easily extended to 128-dimension version without adding much computation complexity.

Binary Descriptors have lower computational complexity. Hence, they are faster than SURF. **Binary Robust Independent Elementary Feature (BRIEF)** [CLO+12] is a general-purpose feature point descriptor which relies on a relatively small number of intensity difference tests to represent an image patch as a binary string. It provides a shortcut and fastest way to find the binary strings directly without finding descriptors. The main advantage of it is the speed and memory requirement. To obtain very good matching results, [CLO+12] shows that 128 or 256 bits are normally enough. The standard BRIEF descriptor is 32-dimensional vector. Unfortunately, it is not rotational invariant. In **ORB** [RRKB11] the orientation of keypoint have been used to orient BRIEF along the keypoint direction to give steered BRIEF, which is rotational invariant and resistant to noise. In **BRISK** [LCS11], orientation of keypoint and pair-wise brightness comparison tests are used to generate 512-bit string that represents 64-dimensional vector. **Fast REtinA Keypoint (FREAK)** [AOV12] is inspired by the human visual system. A retinal sampling pattern is applied around the keypoint, and a 512-bit binary string is computed by comparing the pixel intensities over the sampling pattern using a sequence of DoG.

In our work, we have used SURF descriptor to describe the point features. To improve the speed without compromising that much of performance, we have not consider rotational invariance of SURF descriptor.

b) Line Descriptors

In [WWH09], SIFT like descriptors for line segments known as **Mean Standard Deviation Line Descriptor (MSLD)** is proposed. In this method, the pixel support region (PSR) is defined for each line segment and then the PSR is divided into non-overlapped sub-region. Line Gradient Description Matrix (GDM) is calculated for each sub-region. Mean and Standard deviation of columns of GDM gives MSLD. Finally, vectors are normalized to unit norm in order to cope with illumination changes. 9 sub-regions each with a size of 5×5, results 72-dimensional descriptor. MSLD descriptor is highly distinctive for matching under rotation, illumination change, image blur, viewpoint change, noise, JPEG compression and partial occlusion. However, the computation cost for MSLD is high.

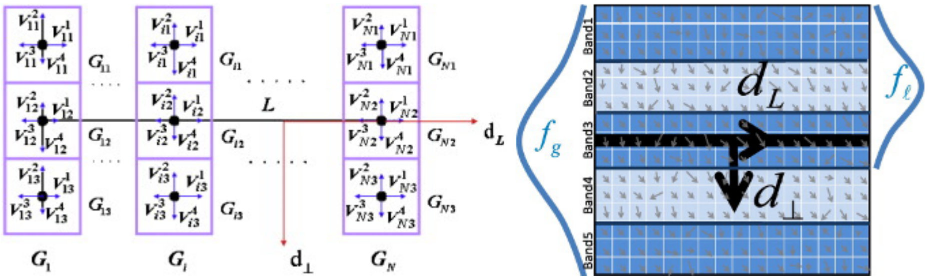


Figure 2.8 – PSR for MSLD [WWH09](left) and LSR for LBD [ZK13] (right).

[BFVG05] describes line segments by **color histogram** obtained from two stripes parallel to the segment, one on either side with 3 pixels distance between the stripes. Color histogram is created with 18 bins for Hue, 3 for Saturation, 3 for Value, and additional four grey levels including black and white. However, this method heavy relies on color rather than on texture, which makes this method not useful where color feature is not distinctive, such as in gray images or remote sensing images. [WNY09] uses **line signatures** to describe the lines. With multi-scale line extraction strategy, the repeatability of line signatures is improved However, this method is quite computationally expensive. MSLD [WWH09] and color histograms [BFVG05] can achieve scale invariance by the method proposed by [VTVG14].

Line Band Descriptors (LBD), the faster version of MSLD, is used by [ZK13] to describe candidate line. In this method, LBD is computed from the line support region (LSR), which is a local rectangular region centered at the line. This LSR is divided into a set of bands parallel with the line. From the projected gradient of pixels in each band in the line direction and orthogonal direction, Band Description Matrix (BDM) is computed. The mean and standard deviation of columns of BDM gives LBD. Furthermore, to reduce the influence of non-linear illumination changes, the value of each dimension of LBD is restrained such that it is smaller than a threshold (0.4 is empirically found to be a good value). Finally, the restrained vector is normalized to get a unit LBD. LSR with 9 bands each of 7 pixels width results in a 72-dimensional descriptor. The scale invariance is archived though detection of keylines in scale rather than making descriptor scale-invariant unlike [VTVG14].

In our work, we have used LBD to describe lines because of its superior performance compromising the speed and accuracy in matching.

2.1.2.4 Feature Correspondences

Feature correspondence aims in ascertaining which features (points, lines, regions) of one image correspond to which features of another image. There are two main approaches to finding features and their correspondences. The first is to find features in one image that can be accurately tracked using a local search technique such as correlation or least squares or using optical flow. The second is to independently detect features in all the images under consideration and then match features based on their local appearance. The former approach is more suitable when images are taken from nearby viewpoints or in rapid sequences, while the latter is more suitable when a large amount of motion or appearance change is expected. Since an image consists of large number of features, it is necessary to have some metrics to rank the potential features. This is normally done by calculating distance between two feature vectors. The common one is Euclidean distance (L-2 norm) for non-binary features. The other distances that can be used are city block distance (L-1 norm), Mahalanobis distance, Minkowski distance [HZ03] etc. For binary features, hamming distance is used. The best candidate match for each feature can be found by identifying feature which has minimum distance. Such feature is known as the nearest neighbor. However, many features from an image may not have any correct match because of background clutter or not detected in other image. In order to discard features that do not have any good match to the database, the simplest idea is to set a global threshold (maximum distance) and to return all matches from other images within this threshold. However, setting the threshold too high results in too many incorrect matches being returned (false positive) whereas setting the threshold too low results in too many correct matches being missed (false negatives). Hence more effective measure is obtained by comparing the distance of the closest neighbor to that of the second-closest neighbor (nearest neighbor distance ratio). Rejecting the matches that have ratio greater than certain threshold will reject many false matches [Low04]. Additionally, cross test can be employed to remove false candidate. To perform the quick matching of high dimensional in presence of huge features, two algorithms have been found to be the most efficient [ML14]: Randomized k-d Tree Algorithm and The Priority Search K-Means Tree Algorithm. Both of these algorithm are available as the Fast Library for Approximate Nearest Neighbors (FLANN) [ML14]. Similarly, the algorithms like Locality Sensitive Hashing (LSH) and parallel searching of randomized hierarchical trees are available for quick matching with binary features [ML12].

The alternative way of establishing correspondences in the image sequence is via tracking that exploits local image properties and kinematics constraints. Tracking is essentially useful and faster way to establish correspondences when the motion between the images are small. The feature point tracking problem is basically a motion correspondence problem under the general assumptions of: indistinguishable points, smooth motion, limited speeds and short occlusions [CV99]. Local motion estimations are based on optical flow or patch matching, which is based on local brightness constancy assumption. Most of the optical flow methods available in the literature are derivatives or improved version of pioneer works of [LK$^+$81] and [HS81]. The state of art methods in optical flow is available in Middlebury website [1]. Patch matching are basically based on registering small image patches by minimizing SSD or SAD, or maximizing NCC or MI.

[DF90, NPVCL08] have presented tracking of line segments based on tracking of each line primitives like orientation of the line segment, its length, the distance of the origin to the line segment and the distance along the line from the perpendicular intersection to the midpoint of the segment using Kalman Filters. [CK98] has used hierarchical Lukas-Kanade optical flow technique to predict the line segments in next frame and line direction attribute defined from the average of all edge directions in a line segment to discriminate closely spaced line segments. Then, local matching is done by a similarity function based on distance from middle line and overlapping ratio. [DRMS07] have used Extended Kalman Filter (EKF) to track lines. [ZK11] have used Nearby Line Tracking to track lines. All of these trackers are prone to errors (depends upon the edge extraction) and not efficient when there are large numbers of lines.

In the line matching side, the method proposed by [SZ97] requires known epipolar geometry that can be computed easily from point correspondences [HZ03]. For short range motion, the matching score is computed based on average cross-correlation of the points common to both lines in two views taken into consideration with the help of epipolar geometry. For long range motion, the correlation path is corrected by a projective warping using a homography computed from the fundamental matrix before calculating the score. However, this method is slow for real time requirements. [FWH10] uses point correspondences for the line matching. Line matching is done through line-point invariant: affine invariant from 2-points and 1-line correspondences or projective invariant from 4-points and 1-line correspondences. The method is fast and works even with some mismatches in point correspondences. Fast matching requires only tentative point correspondences in the neighborhood of the line. This similarity is based on the geometry configuration between lines and points, thus it is robust to many changes in image, including scale, rotation,

[1] http://vision.middlebury.edu/flow/

brightness, viewpoint changes and occlusion. However, the performance of this method heavily depends upon point correspondences that limits its applicability in wide range of environment. [ZK13] presented fast line matching scheme using local appearance and geometric constraints. The local appearance matching is done by using LBD to get candidate matches. Then, relational graph is built for candidate matches using geometric attributes: intersection ratios, projection ratios, and angle between candidate matches from line gradients. Finally, graph matching is done to get matched lines. This method is fast and works well for low textured scene but the matching process is not affine invariant.

In our work, we have used matching of SURF points with FLANN algorithm with L-2 norm as distance metrics. Further, we have used cross check and nearest neighbor distance ratio test to remove outliers. For tracking of FAST corners, we have used modified version of KLT tracker with isotropic scaling and contrast correction as mentioned in [DSRC11]. For line matching, we have used the method provided by [ZK13], which is based on ED Lines and LBD.

Still, the matched feature sets contain outliers that can be filtered by statistically robust methods like RANSAC (RANdom SAmple Consensus) while estimating the geometric transformation or homography, fundamental/essential matrix or tensors [HZ03].

2.1.3 Geometry of Multiple Views

Multiple view geometry is the projective geometry between two or more views. It is independent of scene structure, and only depends on the intrinsic parameters and relative pose of the cameras. If $\boldsymbol{w}$ denotes some matrix/vector in first view, then $\boldsymbol{w}'$ and $\boldsymbol{w}$” denotes the same but for second and third view respectively.

2.1.3.1 The Epipolar Constraint

The epipolar geometry gives the fundamental geometric relationship between two views. This relationship is encapsulated in a unique 3×3, rank 2 homogeneous matrix called Fundamental Matrix. If 3D point $\boldsymbol{X}$ has image correspondences $\boldsymbol{x}$ and $\boldsymbol{x}'$ in two views ($\boldsymbol{x} \leftrightarrow \boldsymbol{x}$') and $\boldsymbol{F}$ is the fundamental matrix, then the image points satisfy the following relationship

$$\boldsymbol{x}'^T \boldsymbol{F} \boldsymbol{x} = 0. \tag{2.11}$$

$\boldsymbol{F}$ has seven degrees of freedom because the common scaling of homogeneous matrix is not significant and determinant of $\boldsymbol{F}$ is zero (rank 2). Fundamental matrix maps point to line between the views. As shown in Fig. 2.9, a point in the first image $\boldsymbol{x}$ defines a line in the second $\boldsymbol{l}' = \boldsymbol{F}\boldsymbol{x}$, which is the epipolar line of $\boldsymbol{x}$. If $\boldsymbol{l}$ and $\boldsymbol{l}'$ are corresponding epipolar

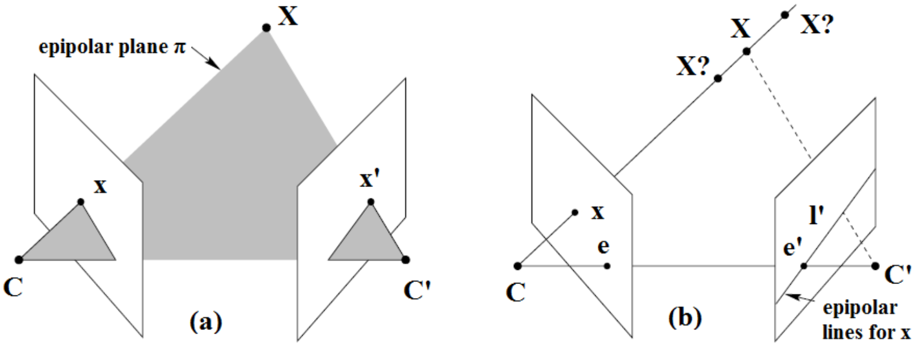


Figure 2.9 – Epipolar geometry between two views. Taken from [HZ03].

lines, then any point $\boldsymbol{x}$ on $\boldsymbol{l}$ is mapped to the same line $\boldsymbol{l}'$. The point of intersection of the line joining the optical centers with the image plane is known as epipole. The epipole is the image of the optical center of the other camera. For any point $\boldsymbol{x}$, except $\boldsymbol{e}$, the epipolar line $\boldsymbol{l}' = \boldsymbol{F}\boldsymbol{x}$ contains the epipole $\boldsymbol{e}'$. The epipole in the left image is $\boldsymbol{e}$, which is the right null-space of $\boldsymbol{F}$ i.e. $\boldsymbol{F}\boldsymbol{e} = \boldsymbol{0}$. Similarly, $\boldsymbol{e}'$ is epipole in the right image, which is the left null-space of $\boldsymbol{F}$ i.e. $\boldsymbol{e}'^T\boldsymbol{F} = 0$.

If the intrinsic parameters of the cameras ($\boldsymbol{K}$ and $\boldsymbol{K}'$) are known, the image points can be expressed in normalized coordinates as

$$\hat{\boldsymbol{x}} = \boldsymbol{K}^{-1}\boldsymbol{x} \text{ and } \hat{\boldsymbol{x}'} = \boldsymbol{K}'^{-1}\boldsymbol{x}'. \tag{2.12}$$

The Essential matrix $\boldsymbol{E}$ defines the the geometric relationship between $\hat{\boldsymbol{x}} \leftrightarrow \hat{\boldsymbol{x}'}$ as

$$\hat{\boldsymbol{x}'}^T\boldsymbol{E}\hat{\boldsymbol{x}} = 0. \tag{2.13}$$

$\boldsymbol{E}$ has five degrees of freedom (three for rotation and two for the direction of translation – the magnitude of translation cannot be recovered due to the depth/speed ambiguity) and has two constraints: (1) its determinant is zero, and (2) its two non-zero singular values are equal. From (2.11-2.13), we have following relationship between $\boldsymbol{E}$ and $\boldsymbol{F}$.

$$\boldsymbol{E} = \boldsymbol{K}'^T\boldsymbol{F}\boldsymbol{K}. \tag{2.14}$$

If $\boldsymbol{R}$ and $\boldsymbol{t}$ are relative rotation and translation between two cameras, the camera matrices $\boldsymbol{P}$ and $\boldsymbol{P}'$ can be written as

$$\boldsymbol{P} = \boldsymbol{K}[\boldsymbol{I}\,|\,\boldsymbol{0}] \text{ and } \boldsymbol{P}' = \boldsymbol{K}'[\boldsymbol{R}\,|\,\boldsymbol{t}].$$

The Essential matrix can be written as

$$\boldsymbol{E} = [\boldsymbol{t}]_\times\boldsymbol{R} = \boldsymbol{R}[\boldsymbol{R}^T\boldsymbol{t}]_\times. \tag{2.15}$$

This epipolar geometry is used later in our work in Chapter 7 for geometric verification of matched/tracked points to remove outliers with RANSAC.

2.1.3.2 The Homography Constraint

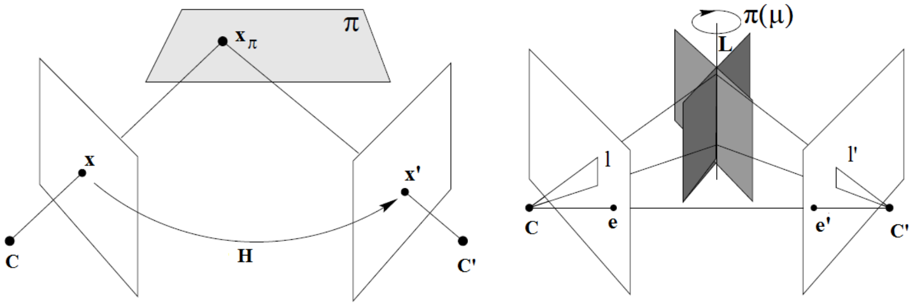


Figure 2.10 – Homography. Taken from [HZ03].

Homography defines the relations between any two images of same planar surface $\boldsymbol{\pi}$ in the space as shown in Fig. 2.10. If $\boldsymbol{n}$ is the normal of the plane and d is the perpendicular distance of plane from first camera center, the 3×3 homography matrix $\boldsymbol{H}$ is given by the following relation [HZ03]

$$\mathbf{H} = \boldsymbol{K}'(\boldsymbol{R} - \frac{\boldsymbol{t}\mathbf{n}^T}{d})\boldsymbol{K}^{-1}., \tag{2.16}$$

The direct mapping between the corresponding points $\boldsymbol{x} \leftrightarrow \boldsymbol{x}$' by homography is given as

$$\boldsymbol{x}' \simeq \boldsymbol{H}\boldsymbol{x}. \tag{2.17}$$

This relations hold true only up-to a scale. A line in 3-space is defined by intersection of two planes, which lies on a pencil of planes as shown in Fig. 2.10(right). This pencil of planes induces a pencil of homographies between the two images, any member of this family will map the corresponding lines to each other. If $\boldsymbol{l} \leftrightarrow \boldsymbol{l}$' are corresponding lines, then their relation defined by homography is given as

$$\boldsymbol{l}' \simeq \boldsymbol{H}^{-T}\boldsymbol{l}. \tag{2.18}$$

2.1.3.3 The Trifocal Constraint

The trifocal geometry gives the fundamental geometric relationship between three views like fundamental matrix for two views. It is basically $3 \times 3 \times 3$ matrix with 18 degrees of freedom. Let set of three matrices $\{\boldsymbol{T}_1, \boldsymbol{T}_2, \boldsymbol{T}_3\}$ constitute the trifocal tensor, where each matrix is 3×3. Then trifocal tensor can be represented by $[\boldsymbol{T}_1, \boldsymbol{T}_2, \boldsymbol{T}_3]$, or more

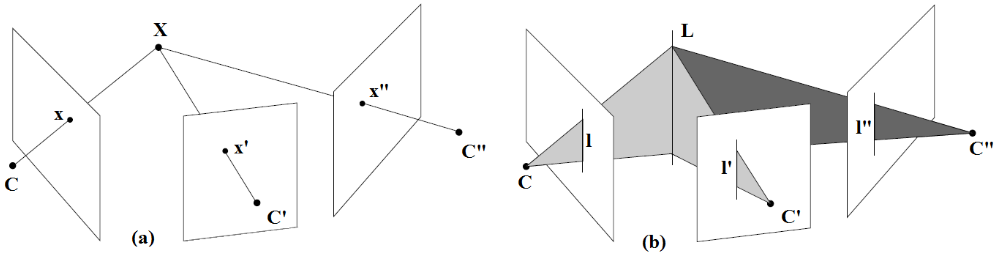


Figure 2.11 – Trifocal Tensor. Picture taken from [HZ03].

briefly by $[\boldsymbol{T}_i]$ [HZ03]. If $\boldsymbol{x} \leftrightarrow \boldsymbol{x}' \leftrightarrow \boldsymbol{x}"$ are point correspondences and $\boldsymbol{l} \leftrightarrow \boldsymbol{l}' \leftrightarrow \boldsymbol{l}"$ are line correspondences in three views as shown in Fig. 2.11, then there is following trifocal constraint [HZ03]

$$\begin{gathered}[\boldsymbol{x}']_\times (\sum_i \boldsymbol{x}^i \boldsymbol{T}_i)[\boldsymbol{x}"]_\times = \boldsymbol{0}_{3\times3}. \\ (\boldsymbol{l}'^T[\boldsymbol{T}_1, \boldsymbol{T}_2, \boldsymbol{T}_3]\boldsymbol{l}")[\boldsymbol{l}]_\times = \boldsymbol{0}^T \text{ or } \boldsymbol{l}'^T[\boldsymbol{T}_1, \boldsymbol{T}_2, \boldsymbol{T}_3]\boldsymbol{l}" = \boldsymbol{l}^T.\end{gathered} \tag{2.19}$$

Trifocal tensor unlike epipolar geometry gives the geometric relations between lines but in three views. This property is used later in our work for geometric verification of matched lines to remove outliers with RANSAC in Chapters 6 and 7. Trifocal tensor can be computed from at least 6 point correspondences [HZ03] or 9 line correspondences [KOA14] in 3 views. From trifocal tensor, we can easily extract fundamental matrix between two views and camera matrices. Let $\boldsymbol{e}'$ and $\boldsymbol{e}"$ be the epipoles and $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ be the left and right null-vectors respectively of $\boldsymbol{T}_i$, i.e. $\boldsymbol{u}_i^T \boldsymbol{T}_i = \boldsymbol{0}^T$, and $\boldsymbol{T}_i \boldsymbol{v}_i = 0$. Then, the epipoles can obtained from the null space of following 3×3 matrices inside $[..]$: $\boldsymbol{e}'[\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3] = 0$ and $\boldsymbol{e}"^T[\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3] = 0$. Now, the fundamental matrices between first and second ($\boldsymbol{F}_{21}$), and first and third ($\boldsymbol{F}_{31}$) views can be obtained as

$$\boldsymbol{F}_{21} = [\boldsymbol{e}']_\times [\boldsymbol{T}_1, \boldsymbol{T}_2, \boldsymbol{T}_3]\boldsymbol{e}" \text{ and } \boldsymbol{F}_{31} = [\boldsymbol{e}"]_\times [\boldsymbol{T}_1^T, \boldsymbol{T}_2^T, \boldsymbol{T}_3^T]\boldsymbol{e}'.$$

With $\boldsymbol{e}'$ and $\boldsymbol{e}"$ normalized to unit norm, the camera matrices $\boldsymbol{P}$, $\boldsymbol{P}'$ and $\boldsymbol{P}"$ can be obtained as

$$\boldsymbol{P} = [\boldsymbol{I} \,|\, \boldsymbol{0}],\ \boldsymbol{P}' = [[\boldsymbol{T}_1, \boldsymbol{T}_2, \boldsymbol{T}_3]\boldsymbol{e}" \,|\, \boldsymbol{e}'], \text{ and } \boldsymbol{P}' = [(\boldsymbol{e}"\boldsymbol{e}"^T - \boldsymbol{I})[\boldsymbol{T}_1, \boldsymbol{T}_2, \boldsymbol{T}_3]\boldsymbol{e}' \,|\, \boldsymbol{e}"].$$

2.1.3.4 Robust Estimation by RANSAC

RANSAC is an iterative method for robust fitting of models to data that can contain outliers. The basic algorithm for RANSAC is given as [FB81]

1. Randomly select m (minimum number of data required to estimate the parameters) data samples and estimate the parameters of the given model.

2. Find how many data items fit the model with parameters within a user given tolerance.
3. If number of inliers is big enough, re-estimate the model will all inliers and exit with success.
4. Repeat steps 1-4, for N number of times.
5. After N trials, the largest consensus set is selected and the model is re-estimated using all the points in that subset if possible.

If p (usually set to 0.99) is the probability that at least one of the sets of random samples does not include an outlier and w is the probability that any selected data point is an inlier, then K can be calculated as

$$N = \frac{log(1-p)}{log(1-w^m)}. \tag{2.20}$$

From (2.20) it is clear that if m is small, less number of iterations are required to get desired consensus set. In order to remove outliers (incorrect matches), we can use epipolar geometry for two views and trifocal geometry for three views can be used as model. The output of the process (if succeed) is a set of correct matches or inliers. Using correct matches is essentially important when we estimate 3D structure and pose from image correspondences.

a) Geometrical Verification of Point Correspondences in two views with Epipolar Geometry + RANSAC

In two views, epipolar geometry provides strong constraint for the point correspondences. Hence, to verify the point correspondences, we use epipolar constraint as a model to Fundamental matrix $(\boldsymbol{F})$ / Essential matrix $(\boldsymbol{E})$. The process of estimating $\boldsymbol{F}$ or $\boldsymbol{E}$ from point correspondences is described in Sect. 2.1.4.1. The error measure that can be used is geometric error, which is simply a re-projection error. In practice, the first order approximation of geometric error, also known as Sampson distance, is used [HZ03], which is given below

$$e = \sum_i \frac{(\boldsymbol{x}_i'^T \boldsymbol{F}_i)^2}{(\boldsymbol{F}\boldsymbol{x}_i)_1^2 + (\boldsymbol{F}\boldsymbol{x}_i)_2^2 + (\boldsymbol{F}^T\boldsymbol{x}_i')_1^2 + (\boldsymbol{F}^T\boldsymbol{x}_i')_2^2},$$

where$(\boldsymbol{F}\boldsymbol{x})_j^2$ is the square of the $j-th$ entry of the vector $\boldsymbol{F}\boldsymbol{x}$.

b) Geometrical Verification of Line Correspondences in three views with Trifocal Geometry + RANSAC

For two views, the line segments do not provide strong geometric constraints as opposite to points. The trifocal tensor provides a strong geometric constraint for lines, but it requires line correspondences in three views. From trifocal constraint $\boldsymbol{l}'^T[\boldsymbol{T}_1,\boldsymbol{T}_2,\boldsymbol{T}_3]\boldsymbol{l}")[\boldsymbol{l}]_\times = \mathbf{0}^T$, we can obtain linear equation in the entries of Trifocal Tensor $\boldsymbol{T}$ ($\boldsymbol{t}$) as

$$\boldsymbol{l}_p\boldsymbol{l}'_q\boldsymbol{l}"_r\varepsilon^{piw}\boldsymbol{T}^i_{qr} = \mathbf{0}^w, \tag{2.21}$$

$$\text{or } \boldsymbol{At} = 0. \tag{2.22}$$

where the subscripts denotes the individual element of line vector or tensor matrix and $\varepsilon^{piw}$is given as

$$\varepsilon^{piw} = \begin{cases} 0 & \text{if p, i, w are not distinct} \\ +1 & \text{if piw is an odd permutation of 123} \\ -1 & \text{if piw is an even permutation of 123} \end{cases}.$$

(2.21) gives 3 set of equations. However, only 2 are linearly independent. So, we need at least 13 line correspondences to compute trifocal tensor. The solution of equation $\boldsymbol{At} = 0$ gives trifocal tensor. The endpoints of lines are normalized as described in [Har97] before computing line vectors. The error function for minimizing geometric error for RANSAC is approximated as [HZ03]

$$e = \sum_i \boldsymbol{\varepsilon}_i^T(\boldsymbol{J}_i\boldsymbol{J}_i^T)^{-1}\boldsymbol{\varepsilon}_i,$$

where $\boldsymbol{\varepsilon}_i = \boldsymbol{A}_i\boldsymbol{t}$ is an algebraic error corresponding to a single 3-view correspondence and $\boldsymbol{J}$ is the matrix of partial derivatives of $\boldsymbol{\varepsilon}$.

2.1.4 Structure from Motion (SfM)

Structure from Motion is the process of reconstructing 3D scene and estimating 6DOF pose from unordered sets. With no constraints on the camera calibration matrix or on the scene, we get a projective reconstruction. Hence, we need additional information to upgrade the reconstruction to Metric or Euclidean. The process generally involves finding the correspondences (points) between the scene and use multi-view geometry to recover 3D scene and pose. Additional bundle adjustment [TMHF99] steps are employed for refining SfM by minimizing re-projection error.

2.1.4.1 SfM for Two Views

For uncalibrated two views, at least 7 points are required for SfM. For calibrated case, five points are sufficient. For the calibrated case, the simplest process involves estimation of essential matrix, decomposition of $\boldsymbol{E}$ into $\boldsymbol{R}$ and $\boldsymbol{t}$, triangulation of 2D points to get 3D scene.

I) Estimation of Essential Matrix

a) Normalized 8 Point Algorithm: This is the classical method for computing Fundamental matrix/ Essential matrix [HZ03]. Since the fundamental matrix $\boldsymbol{F}$ is 3×3 matrix determined up to an arbitrary scale factor, 8 equations are required to obtain a unique solution. The simplest way to compute the fundamental matrix consists of using epipolar constraint

$$\boldsymbol{x}'^T \boldsymbol{F} \boldsymbol{x} = 0 \tag{2.23}$$

where (x, x') are corresponding points of two views. This equation can be rewritten under the following form

$$[xx' \; yx' \; x' \; xy' \; yy' \; y' \; x \; y \; 1]\boldsymbol{F} = 0 \tag{2.24}$$

where $\boldsymbol{F} = [F_{11}\, F_{12}\, F_{13}\, F_{21}\, F_{22}\, F_{23}\, F_{31}\, F_{32}\, F_{33}]^T$ a vector containing the elements of the fundamental matrix $\boldsymbol{F}$. By stacking eight of these equations in a matrix $\boldsymbol{A}$ the following equation is obtained:

$$\boldsymbol{A}\boldsymbol{F} = 0. \tag{2.25}$$

The solution in this case is given by the right singular vector of A associated with the smallest singular value. In practice, the Fundamental matrix obtained does not satisfy the constraint $det(\boldsymbol{F}) = 0$. So, we have to find the best rank two matrix from $\boldsymbol{F}$. It is done by Singular Value Decomposition of $\boldsymbol{F}$ $(= \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T)$ and forcing smallest singular value to be zero. Then the required fundamental matrix is obtained by $(\boldsymbol{U}\boldsymbol{D}'\boldsymbol{V}^T)$. The points are normalized before calculating fundamental matrix and finally transformed back to original coordinate system as described by Hartley [Har97].

After computing the fundamental matrix, the essential matrix can be calculated by

$$\boldsymbol{E} = \boldsymbol{K}^T \boldsymbol{F} \boldsymbol{K}. \tag{2.26}$$

Instead, we can use normalized image points to calculate essential matrix directly from (2.23-2.25). In this case, the smallest singular value of obtained $\boldsymbol{E}$ is forced to zero and the remaining two singular values are set to 1 to satisfy the property of Essential matrix.

b) Nister's 5 point Algorithm

This is used to estimate relative pose of two fully-calibrated cameras from five image point correspondences (minimal case). The 5-point method is essentially unaffected by the planar degeneracy and still works. It also works well for sideways motion also[Nis04]. Consider a camera, with constant intrinsic matrix $\boldsymbol{K}$, observing a static scene. Two corresponding image points $\boldsymbol{x}$ and $\boldsymbol{x}'$ are then related by Essential matrix $\boldsymbol{E}$ by

$$\boldsymbol{x}'\boldsymbol{E}\boldsymbol{x} = 0 \tag{2.27}$$

Besides, $\boldsymbol{E}$ also satisfies determinant zero constraint

$$det(\boldsymbol{E}) = 0 \tag{2.28}$$

and cubic constraints due to equal singular values which is also known as trace constraint.

$$\boldsymbol{E}\boldsymbol{E}^T\boldsymbol{E} - \frac{1}{2}trace(\boldsymbol{E}\boldsymbol{E}^T)\boldsymbol{E} = 0 \tag{2.29}$$

Overview of 5-point Algorithm

1. Writing down the epipolar equation 2.27 for the five points, we can get a null-space representation $\boldsymbol{E} = a\boldsymbol{E_1} + b\boldsymbol{E_2} + c\boldsymbol{E_3} + w\boldsymbol{E_4}$,where, $\boldsymbol{E}_i, i = 0,1,2,3$ are the null-space bases. Using the fact that $\boldsymbol{E}$ is homogeneous, without loss of generality it can be set $w = 1$.

2. Using the nine equations of equation 2.29, a 10×20 coefficient matrix corresponding to a monomial vector can be obtained as
$$[a^3, b^3, a^2b, ab^2, a^2c, a^2, b^2c, b^2, abc, ab, ac^2, ac, a, bc^2, bc, b, c^3, c^2, c, 1].$$

3. Then Grobner basis is computed, which is similar to performing a Gauss-Jordan elimination on the 10×20 matrix. After that 10×10 action matrix for multiplication by one of the unknowns is computed. This is as simple as extracting the correct elements from the eliminated 10×20 matrix and organizing them to form the action matrix. (The details are available in [SEN06]).

4. The left eigenvectors of the action matrix are computed. The real values for the three unknowns at all of the solution points are read off and back-substituted to obtain the solutions for the essential matrix.

In our work, we have used 5-point algorithm to estimate Essential matrix for removing outliers in point using RANSAC.

II) Recovering Structure and Motion from Essential Matrix

Now, matrix $\boldsymbol{E}$ can be easily decomposed to obtain $\boldsymbol{R}$ and $\boldsymbol{t}$. In general, four solutions are possible. The correct one is obtained by imposing cheirality constraint (reconstructed point should lie infront of the camera). After that we can easily obtain projection matrices for both cameras. With projection matrices and 2D image correspondences, we can easily obtain 3D location via triangulation. These methods can be easily found in standard computer vision textbooks [HZ03, Sze10].

Given the Rotation and Translation $[\boldsymbol{R}_i|\boldsymbol{t}_i]$ of the camera w.r.t. first camera at $[\boldsymbol{I}|\boldsymbol{0}]$, the vector denoting the camera position in a world coordinate frame $(\boldsymbol{c}_i)$ can be calculated as

$$\boldsymbol{c}_i = -\boldsymbol{R}_i^T \boldsymbol{t}_i. \tag{2.30}$$

The actual position in the world coordinate frame can only be calculated if the scale factor is known. This scale factor can be determined if the knowledge of robot movement is known like odometry information or in case of stereo vision.

2.1.4.2 SfM for Multiple Views

In previous subsections, we can see how structure and motion can be estimated from two views of same scene. What if we have more views. They can be used for verification. If camera calibration is known, the pair of stereo can be used incrementally to obtain 3D structure and motion. With no constraints on the camera calibration matrix or on the scene, we get a projective reconstruction. We need additional information to upgrade the reconstruction to affine, similarity, metric or Euclidean [HZ03]. In order to minimize re-projection error for refining structure and motion, the techniques like bundle adjustment [TMHF99] is used. There also exists various techniques which first perform camera self-calibration (recovery which camera motion and intrinsic parameters) using rigidity of the scene and then readily calculates structure from the scene from minimum 3 views [Pol99, PKVVG00].

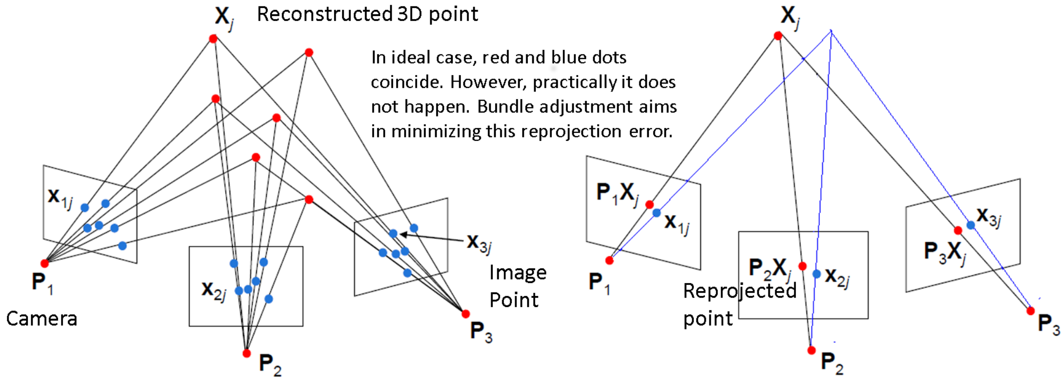


Figure 2.12 – Multiple view SfM.

2.1.5 Visual SLAM

Visual SLAM [FPRARM15] can be seen as a special case of SfM. It refers to process of constructing or updating a map of an unknown environment while simultaneously keeping track of a location within it using vision. SLAM [DWB06, BDW06] consists of multiple parts: landmark extraction or feature detection, data association or feature matching, state estimation or 3D pose estimation, state update or update to global pose and pose correction, and landmark update or incorporation of new features in the map. SLAM is hard, because a map is needed for localization and a good pose estimate is needed for mapping. Hence, SLAM is a chicken-or-egg problem. The SLAM problem in robotics is basically to find map of features and path of the robot from robot's controls and relative observations.

In SLAM, errors in map and pose estimates are correlated. So, they need to be corrected so that global consistency of the trajectory and the map can be obtained. This global consistency can be generally obtained using optimizations like Bundle Adjustment [TMHF99] or Pose-Graph Optimization [GKSB10]. Loop constraints are very valuable constraints for such optimizations, which can be obtained by evaluating visual similarity between the current camera images and past camera images. This visual similarity can be obtained via feature matching. This already known information can be used to refine map and pose. This process of re-recognizing already mapped area or previously visited location is known as Loop Closure. The work of [ESC14] performs SLAM by direct method without feature detection and matching. Although metric SLAM is the dominant in the state of art, recent works like [MW10, GMMW10, CN11, MMW12] performs SLAM upon appearance in which loop closing is generally performed in the topological space.

2.1.6 Visual Odometry

Visual Odomety (VO) is a particular case of SfM that focuses on estimating the 3D motion of the camera sequentially (as a new frame arrives) and in real time [SF11, FS12]. So, VO only aims to the local consistency of the trajectory unlike SLAM that aims to the global consistency of the trajectory and of the map. Hence, VO is SLAM before closing the loop. VO trades off consistency for real-time performance, without the need to keep track of all the previous history of the camera. In practice only few last poses are used to refine pose via local bundle adjustment or local pose graph optimization.

2.2 Mobile Robots

Mobile robots are the robots having locomotion capability. The locomotion may be manual remote or tele-operated, semi-autonomous and autonomous. The locomotion may be legged as in humanoid robots, wheeled as in autonomous ground vehicles, free-floating as in marine robots or crawling locomotion as in snake robots. There are also the robots that can fly, known as aerial robots. For the autonomous robots, it must require some mechanism to sense the environment. Vision sensors are one of the most widely used sensors to perceive the environment. However, other sensors like sonars, ultrasounds, tactile, Light Detection And Ranging (LIDAR) etc. exists to sense the environment. Throughout the thesis unless otherwise stated, we basically consider wheeled autonomous mobile robots using camera as only sensing device.

2.2.1 Navigation of Mobile robots

Navigation is a fundamental problem in mobile robotics. For the robot to navigate autonomously, it has to answer three questions [LDW91]: i) Where am I?, ii) Where am I going?, and iii) How do I get there? These questions can be restated as localization, mapping or determining the goal, and path planning problems respectively. Localization is the process of estimating the current position of the robot relatively to some model of the environment using sensor measurements. Such estimation normally involves measurement, correlation and triangulation. The mapping problem exists when the robot does not have a map of its environment. Maps can be built in advance or incrementally build one as robot navigates (like in SLAM). Sometimes, navigation can be done recognizing special landmarks of the environment (e.g. vanishing point of corridor guidelines as in [FOPV13]) without using the map. Path planning involves finding the optimum path from the current position to its goal considering the obstacles. The cost of planning is proportional to the size and complexity of the environment. Path planning can be either local that uses only needed information of the environment near the robot, or global that use full information of the environment.

2.2.2 Mobile Robots Modeling and Control

Robots consists of a mechanical structures in which a set of rigid bodies, called links, are connected by joints, and actuated by motors in some or all of the joints. These motors (actuators) make the mechanical structure move. Position sensors are possibly connected to the joints to measure the motion of joints. To control a robot, we need to represent the robot's state with some quantifiable variables. If we know the state description, we can model the motion of the robot with differential equations, which is known as kinematics. Once we have the kinematics equations, we can develop a control law that bring a robot to a desired location. In particular, we will concentrate in the mobile robot with differential drive, whose motion is defined through rolling and sliding constraints taking effect at the wheel-ground contact points.

2.2.2.1 Robot Kinematics

Mobile robot kinematics is the dynamic model of how a mobile robot behaves. Mobile robots can move unbound with respect to their environment. It means that there is no direct way to measure the position. Position must be integrated over time, which leads to inaccuracies in the estimation of position and motion. Each wheel contributes to the robot's motion and imposes constraints on the motion. These constraints must be expressed w.r.t. the reference frame (robot frame or local coordinate frame).

The robot reference frame ($\mathcal{F}_R$) is three dimensional including position on the plane and

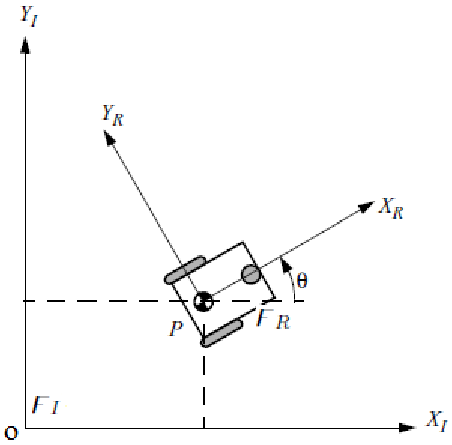


Figure 2.13 – The global reference frame and the robot local reference frame [SNS11].

the orientation as shown in Fig. 2.13, where axes$\{X_R, Y_R\}$ define such reference frame relative to point P on the robot chassis and the axes $\{X_I, Y_I\}$ define inertial global reference frame ($\mathcal{F}_I$) with origin O. If the position of P in $\mathcal{F}_I$ is specified by coordinates x and y and the angular difference between $\mathcal{F}_I$ and $\mathcal{F}_R$ is θ, the pose and the velocity of the robot can be defined as a vector with these three elements. Let $\xi_\mathbf{I}$ be the position and $\dot{\xi}_\mathbf{I}$ be the motion of robot in $\mathcal{F}_I$. Then we have,

$$\xi_\mathbf{I} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \text{ and } \dot{\xi}_\mathbf{I} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}, \tag{2.31}$$

where $\dot{x}$, $\dot{y}$, and $\dot{\theta}$ represent velocity of x, y and θ respectively in $\mathcal{F}_I$. The orthogonal rotation matrix ($\mathbf{R}(\theta)$) is used to map position and motion in $\mathcal{F}_I$ to that in the $\mathcal{F}_R$, which is given as

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.32}$$

If $\xi_\mathbf{R}$ is the position and $\dot{\xi}_\mathbf{R}$ is the motion of robot in $\mathcal{F}_R$, then we can write

$$\xi_{\mathbf{R}} = \mathbf{R}(\theta)\xi_{\mathbf{I}} = \begin{bmatrix} x\cos\theta + y\sin\theta \\ -y\sin\theta + x\cos\theta \\ \theta \end{bmatrix} \text{ and } \dot{\xi_{\mathbf{R}}} = \mathbf{R}(\theta)\dot{\xi_{\mathbf{I}}} = \begin{bmatrix} \dot{x}\cos\theta + \dot{y}\sin\theta \\ -\dot{y}\sin\theta + \dot{x}\cos\theta \\ \dot{\theta} \end{bmatrix}. \tag{2.33}$$

From (2.33), it is clear that the robot will rotate with the same speed with respect to $\mathcal{F}_R$ as $\mathcal{F}_I$. However, the linear velocities are a combination of the velocities with respect to $\mathcal{F}_I$.

The differential drive robot has two wheels, each with diameter d. The contribution of motion of both wheels can be simply added to find the motion of the robot. Let P be in between two wheels so that each wheel is at distance l from P. If $\dot{\phi}_1$ and $\dot{\phi}_2$ are spinning speed of each wheel, we have

$$\dot{x} = \frac{(\dot{\phi}_1 + \dot{\phi}_2)}{2}, y \dot{=} 0, \text{ and } \dot{\theta} = \frac{d(\dot{\phi}_1 + \dot{\phi}_2)}{2l}. \tag{2.34}$$

Hence, if r, l ,θ, $\dot{\phi}$ and $\dot{\phi}_2$ are known, we can estimate the motion in $\mathcal{F}_I$ as

$$\dot{\xi_{\mathbf{I}}} = \mathbf{R}(\theta)^{-1} \begin{bmatrix} \frac{(\dot{\phi}_1+\dot{\phi}_2)}{2} \\ 0 \\ \frac{r(\dot{\phi}_1+\dot{\phi}_2)}{2l} \end{bmatrix} = f(l, r, \theta, \dot{\phi}_1, \dot{\phi}_2) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}, \tag{2.35}$$

where $\mathbf{R}(\theta)^{-1}$ is inverse of $\mathbf{R}(\theta)$. In order to determine the space of possible motions for each robot chassis design, we should consider constraints on robot motion imposed by each wheel. From (2.35), we can find the motion of a robot given its component wheel speeds. This type of kinematic modeling is known as Forward Kinematics. Standard conventions, such as the Denavit-Hartenberg parametrization, exist to evaluate the direct kinematics [SSVO10].

Forward Kinematics provides transformation from joint space to physical space. But it requires accurate measurement of the wheel velocities over time.

Inverse Kinematics provides transformation from physical space to joint space. It helps to estimate the control parameters (wheel velocities) that will make the robot move to a new pose from its current pose. So, it is required for motion control.

Fixed wheels impose non-holonomic constraints i.e. differential equations are not integrable to the final position. Therefore, in order to calculate the final position of the robot, we need to know how this movement was executed as a function of time in addition to the measure of the traveled distance of each wheel. Therefore, we deal with differential kinematics i.e. transformation between velocities instead of positions.

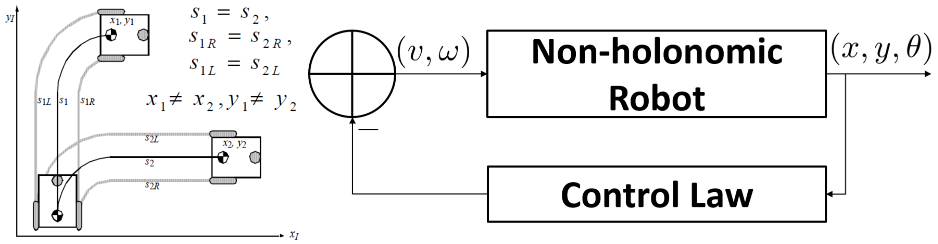


Figure 2.14 – Non-holonomic Robots [SNS11].

2.2.2.2 Robot Motion Control

Robot motion control is the process of controlling the wheel velocities so that the robot moves from current position to desired position. Basically, there are following two strategies of motion control:

a) Open loop control (Trajectory-Following): In this scheme, measured robot position is not fed back for velocity or position control. The path to be followed is divided in motion segments of clearly defined shape straight lines and arcs of a circle. The control problem is thus to pre-compute this trajectory that drives the robot from the initial position to the final position. The scheme looks simple but it has many disadvantages [SNS11]. First, pre-computing a feasible trajectory is not an easy task if the limitations and constraints of the robot velocities and accelerations are considered. Second, the robot does not adapt or correct the trajectory if dynamic changes in the environment occur. Last, the resulting trajectories are usually not smooth because the transitions from one trajectory segment to another are not smooth.

b) Closed loop control (Feedback Control) In this scheme, measured robot position is taken as feedback for velocity or position control. So, the goal of the motion controller is to estimate the robot velocities so that it minimizes the pose error. If v is the forward velocity and ω be the rotational velocity of the robot as shown in Fig. 2.15, clearly from (2.35), we have

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ -\sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \tag{2.36}$$

If ρ is the distance between center of the axle of the wheels P and the final position G, α is the angle between the X_R and $\overrightarrow{PG}$, and β is the angle between the angle between the X_G and $\overrightarrow{PG}$ as shown in Fig. 2.15, from [SNS11] we have

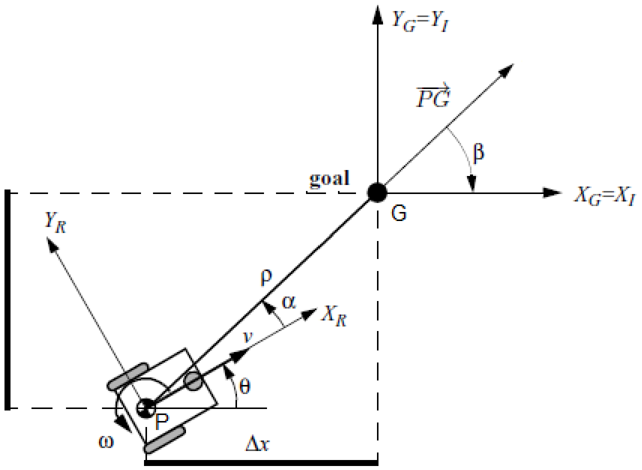


Figure 2.15 – Robot kinematics and its frames of interests [SNS11].

$$\begin{bmatrix}\dot{\rho}\\ \dot{\alpha}\\ \dot{\beta}\end{bmatrix}=\begin{bmatrix}-\cos\alpha & 0\\ \frac{\sin\alpha}{\rho} & -1\\ -\frac{\sin\alpha}{\rho} & 0\end{bmatrix}\begin{bmatrix}v\\ \omega\end{bmatrix}\text{ for I1 and }\begin{bmatrix}\dot{\rho}\\ \dot{\alpha}\\ \dot{\beta}\end{bmatrix}=\begin{bmatrix}\cos\alpha & 0\\ -\frac{\sin\alpha}{\rho} & 1\\ \frac{\sin\alpha}{\rho} & 0\end{bmatrix}\begin{bmatrix}v\\ \omega\end{bmatrix}\text{ for I2,}\tag{2.37}$$

where I1 is the case in which $\alpha \in \left(-\frac{\pi}{2},\frac{\pi}{2}\right]$ and I2 is the case in which $\alpha \in \left(-\pi,-\frac{\pi}{2}\right]\cup\left(\frac{\pi}{2},\pi\right]$.

(2.37) is undefined at $\rho = 0$. Let us define a linear control law as $v = k_\rho \rho$ and $\omega = k_\alpha \alpha + k_\beta \beta$, from (2.37) we obtain

$$\begin{bmatrix}\dot{\rho}\\ \dot{\alpha}\\ \dot{\beta}\end{bmatrix}=\begin{bmatrix}-k_\rho\rho\cos\alpha\\ k_\rho\sin\alpha-k_\alpha\alpha-k_\beta\beta\\ -k_\rho\sin\alpha\end{bmatrix}\text{ for I1 and }\begin{bmatrix}\dot{\rho}\\ \dot{\alpha}\\ \dot{\beta}\end{bmatrix}=\begin{bmatrix}k_\rho\rho\cos\alpha\\ -k_\rho\sin\alpha+k_\alpha\alpha+k_\beta\beta\\ k_\rho\sin\alpha\end{bmatrix}\text{ for I2.}\tag{2.38}$$

The system has a unique equilibrium point at $(\rho,\alpha,\beta) = (0,0,0)$. Thus, it will drive the robot to this point, which is the goal position. In the following sections, we will discuss how vision is used for closed-loop control.

Stability Analysis

To assess the stability of the closed-loop systems, Lyapunov analysis [KG96] is generally used. Let us consider nonlinear system $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ having a point of equilibrium at $\mathbf{x} = \mathbf{0}$ and candidate function $\mathscr{L}(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ such that

$$\begin{cases} \mathscr{L}(\mathbf{x}) = \mathbf{0} & \text{if and only if } \mathbf{x} = \mathbf{0} \\ \mathscr{L}(\mathbf{x}) > \mathbf{0} & \forall \mathbf{x} \neq \mathbf{0}. \end{cases}$$

Then the system is locally stable if

$$\left\{ \dot{\mathscr{L}}(\mathbf{x}) \leq \mathbf{0} \quad \forall \mathbf{x}, \right.$$

and locally asymptotically stable if

$$\left\{ \dot{\mathscr{L}}(\mathbf{x}) < \mathbf{0} \quad \forall x \neq 0. \right.$$

The global asymptotic stability can be ensured if the system is radially unbounded i.e.

$$\left|\left|x\right|\right| \to \infty \Rightarrow \mathscr{L}(x) \to \infty,$$

which means the system converges to equilibrium point independently of the starting pose.

2.2.2.3 Non-holonomic Robot with Perspective Camera

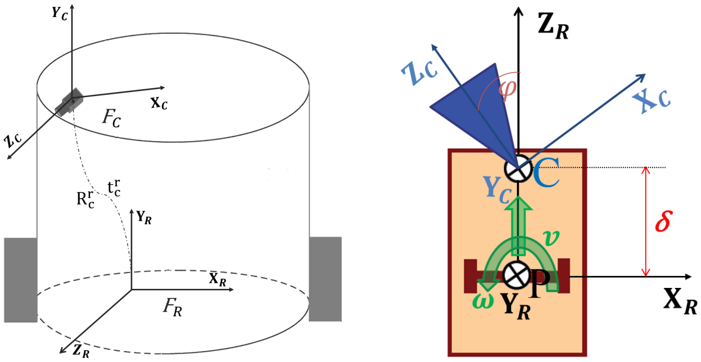


Figure 2.16 – Non-holonomic Robot equipped with perspective camera.

Fig. 2.16 shows the mobile robot equipped with the camera. The measurements obtained using images are on the camera frame $\mathcal{F}_C$. If $\boldsymbol{R}_c^r$ is the rotation and $\boldsymbol{t}_c^r$ is the position vector of origin of the camera frame w.r.t. the robot frame $\mathcal{F}_R$, the coordinate transformation between $\mathcal{F}_C$ and $\mathcal{F}_R$ is given by 4×4 homogeneous transformation matrix $\boldsymbol{H}_c^r$ as

$$\boldsymbol{H}_c^r = \begin{bmatrix} \boldsymbol{R}_c^r & \boldsymbol{t}_c^r \\ 0 & 1 \end{bmatrix}. \tag{2.39}$$

If $\boldsymbol{H}_r^o$ is the transformation between $\mathcal{F}_R$ and $\mathcal{F}_I$, the transformation between $\mathcal{F}_R$ and $\mathcal{F}_I$, $\boldsymbol{H}_c^o$, can be easily obtained as

$$\boldsymbol{H}_c^o = \boldsymbol{H}_r^o \boldsymbol{H}_c^r. \tag{2.40}$$

Similarly, the relationship between camera velocity and robot velocity is given as [SHV06]

$$\begin{bmatrix} 0 \\ 0 \\ v \\ 0 \\ \omega \\ 0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_c^r & [\boldsymbol{t}_c^r]_\times \boldsymbol{R}_c^r \\ \boldsymbol{0}_{3\times 3} & \boldsymbol{R}_c^r \end{bmatrix} \begin{bmatrix} v_{cx} \\ v_{cy} \\ v_{cz} \\ \omega_{cx} \\ \omega_{cy} \\ \omega_{cz} \end{bmatrix}, \tag{2.41}$$

where v_{cx}, v_{cy} and v_{cz} are the linear velocities and ω_{cx},ω_{cy} and ω_{cz} are the rotational velocities of camera in camera frame.

For the camera configuration as shown in Fig. 2.16 (right) i.e. the camera optical axis is orthogonal to the axis of robot rotation (i.e. tilt angle is 0) and the camera center and the robot center of rotation are separated by distance δ. If φ is the pan angle of the camera, the transformation of velocity from robot frame to camera frame is given as:

$$\begin{bmatrix} v_{cx} \\ v_{cy} \\ v_{cz} \\ \omega_{cx} \\ \omega_{cy} \\ \omega_{cz} \end{bmatrix} = \begin{bmatrix} \sin\varphi & -\delta\cos\varphi \\ 0 & 0 \\ \cos\varphi & \delta\sin\varphi \\ 0 & 0 \\ 0 & -1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \tag{2.42}$$

With pan angle 0 i.e. $\varphi = 0$, (2.42) reduces to

$$\begin{bmatrix} v_{cx} \\ v_{cy} \\ v_{cz} \\ \omega_{cx} \\ \omega_{cy} \\ \omega_{cz} \end{bmatrix} = \begin{bmatrix} 0 & -\delta \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & -1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \tag{2.43}$$

2.3 Visual Servoing (VS)

According to the definition given in [CH06], the term Visual Servoing (VS), or visual servo control, refers to the use of computer vision data (extracted from camera images) to control the robot motion in a closed loop. As per this definition, visual servo control relies on techniques from image processing, computer vision, and control theory. The pioneer work is generally considered to be that of [SI73] who describe how a visual feedback loop can be used to correct the position of a robot to increase task accuracy where visual sensing and control are combined in an open-loop fashion. In real visual servoing systems, the control feedback loop is closed around real-time image processing and measurements. The term visual servoing was first used in works of [JA77, Agi79] at SRI International. A large variety of different visual control schemes have been proposed in the literature. The complete overview is presented in classical papers by [Cor93, HHC96], or in more recent ones by [CH06, CH07]. VS is more a general paradigm rather than a specific collection of techniques. Different types of cameras: perspective, catadioptric or generalized cameras, monocular or stereoscopic, or RGB-D, ultrasound probes etc. have been used. Cameras can be mounted either on the robot end-effector (eye-in-hand) or on an external fixed base (eye-to-hand).

The aim of all vision-based control schemes is to minimize an error $\mathbf{e(t)}$, which is typically defined by: [CH06]

$$\mathbf{e(t)} = \mathbf{s(t)} - \mathbf{s}^{\star}, \tag{2.44}$$

where $\mathbf{s(t)}$ is the vector of current features, which may be image coordinates of interest points, current camera poses, etc. The vector $\mathbf{s}^{\star}$ contains the desired values of the features. Once $\mathbf{s}$ is selected, the design of the control scheme can be done by designing a velocity controller. To do this, we require the relationship between the time variation of $\mathbf{s}$ i.e. $\dot{\mathbf{s}}$ and the camera velocity $\mathbf{u_c}$. The camera velocity can be represented by $\mathbf{u_c} = (\mathbf{v_c}, \boldsymbol{\omega}_\mathbf{c})$, where $\mathbf{v_c}$ is the instantaneous linear velocity of the origin of the camera frame and $\omega_\mathbf{c}$ is the instantaneous angular velocity of the camera frame. In spatial frame, $\mathbf{u_c}$ is a 6 element vector which has 3 components of $\mathbf{v_c}$ each representing translational velocity under X, Y and Z axes, and 3 components of $\omega_\mathbf{c}$ each representing rotational velocity around X, Y and Z axes. The relationship between $\dot{\mathbf{s}}$ and $\mathbf{u_c}$ is given as: [CH06]

$$\dot{\mathbf{s}} = \mathbf{L_s u_c}, \tag{2.45}$$

where $\mathbf{L_s} \in \mathbb{R}^{\mathbf{k \times 6}}$ is known as feature Jacobian or interaction matrix related to $\mathbf{s}$ and $k \geq 6$. The relationship between $\mathbf{u_c}$ and time variation of the error $\dot{\mathbf{e}}$ can be obtained from (2.44) and (2.45) as

$$\dot{\mathbf{e}} = \mathbf{L_s}\mathbf{u_c}. \tag{2.46}$$

In order to try to ensure an exponential decoupled decrease of the error (i.e. $\dot{\mathbf{e}} = -\lambda\mathbf{e}$), we control $\mathbf{u_c}$ such that from (2.46) we obtain

$$\mathbf{u_c} = -\lambda\mathbf{L_s^+}\mathbf{e}, \tag{2.47}$$

where $\mathbf{L_s^+}$ is chosen as the Moore-Penrose pseudo-inverse of $\mathbf{L_s}$. In real system, it is not always possible to know perfectly in practice either $\mathbf{L_s}$ or $\mathbf{L_s^+}$. So, an approximation or an estimation of one of these two matrices must be realized. If $\widehat{\mathbf{L_s^+}}$ represents the pseudo-inverse of such estimation or approximation, the control law given by (2.47) becomes

$$\mathbf{u_c} = -\lambda\widehat{\mathbf{L_s^+}}\mathbf{e}. \tag{2.48}$$

This is the basic design implemented by most visual servo controllers. We should select adequate $\mathbf{s}$ for a given task such that the rank of $\mathbf{L_s} = m$, where $m \leq 6$ is the number of controlled camera degrees of freedom and $k \geq m$. Depending upon the nature of features selected, VS can follow one of the approaches discussed below.

2.3.1 Pose-Based Visual Servoing (PBVS)

In PBVS [CH06] systems, features are extracted from an image, and subsequently used to estimate 3-D parameters like pose of the camera w.r.t some reference frame. It is then typical to define $\mathbf{s}$ in terms of the parameterization used to represent the camera pose. After that an error is computed in the metric/Euclidean task space, which is used in visual control. Thus, in PBVS if the pose estimation problem, which is a classical computer vision problem of 3D-localization, is solved, then the control problem is just the classical robotics problem of tracking a trajectory in geometric space. PBVS provides better response to large translational and rotational camera motions. If pose estimation is perfect, then it guarantees global asymptotic stability [CH06]. It is free of image singularities, local minima, and camera-retreat problems. However, PBVS is more sensitive to image noise, camera and object model errors and camera calibration errors.

2.3.2 Image-Based Visual Servoing (IBVS)

In an IBVS [CH06] system the pose estimation is solved implicitly unlike PBVS i.e. the object is in desired relative pose if the current view of the object matches the desired view. The information extracted from the camera (e.g. the position of some key points, or contours) is directly used as feature. The goal configuration is described in terms of the value that the features assume when camera is in the desired pose. Another approach is to directly use intensity information without any feature extraction, where IBVS problem is seen as a case of image registration problem to maximize or minimize some similarity measurement [Dam10]. The advantage of IBVS is that it does not require full pose estimation and hence is computationally more economical than PBVS. The positioning accuracy and closed-loop stability of IBVS is less sensitive to camera-calibration errors and image noise than PBVS. However, IBVS may lead to image singularities that might cause control instabilities. Another issue with IBVS is the camera-retreat problem [Cor11] i.e. for the commanded pure rotations around the optical axis, the camera often moves away from the target in a normal direction and then returns. Moreover, when all 6 dof are considered, the convergence and stability of IBVS is ensured only in a small neighborhood of the desired camera pose. The domain of this neighborhood is analytically impossible to determine. Determining the domain of this neighborhood is still an open issue, even if this neighborhood is surprisingly quite large in practice [CH06].

Now, we present how interaction matrix can be calculated for the point features. Let us consider a 3D point $\mathbf{X} = [X\ Y\ Z]^T$ in the camera frame which projects in the image as a 2D point with coordinates $\mathbf{x} = [x\ y]^T$. (x, y) can be obtained directly from pixel coordinates by normalizing with matrix K (of (2.3)). Then, we have

$$x = \frac{X}{Z} \text{ and } y = \frac{Y}{Z}. \tag{2.49}$$

Taking the time derivative of (2.49), we obtain

$$\dot{x} = \frac{\dot{X} - x\dot{Z}}{Z} \text{ and } \dot{y} = \frac{\dot{Y} - y\dot{Z}}{Z} \tag{2.50}$$

The relation between velocity of $\mathbf{X}$ and v_c is given as $\dot{\mathbf{X}} = -v_c - \omega_c \times \mathbf{X}$, which yields

$$\begin{aligned} \dot{X} &= -v_x - \omega_y Z + \omega_z Y, \\ \dot{Y} &= -v_y - \omega_z X + \omega_x Z \quad \text{and} \\ \dot{Z} &= -v_z - \omega_x Y + \omega_y Z. \end{aligned} \tag{2.51}$$

From (2.50) and (2.51) we obtain

$$\begin{aligned} \dot{x} &= -\tfrac{v_x}{Z} + \tfrac{xv_z}{Z} + xy\omega_x - (1+x^2)\omega_y + y\omega_z \quad \text{and} \\ \dot{y} &= -\tfrac{v_y}{Z} + \tfrac{yv_z}{Z} + (1+y^2)\omega_x - xy\omega_y - x\omega_z, \end{aligned} \tag{2.52}$$

which can be written as $\dot{\mathbf{x}} = \mathbf{L_x u_c}$, where the interaction matrix $\mathbf{L_x}$ related to $\mathbf{x}$ is

$$\mathbf{L_x} = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix}. \tag{2.53}$$

To compute $\boldsymbol{L_x}$, we need to know or approximate Z. To control 6 DOF, at least 3 points are necessary ($k \geq 6$). Let it be $\mathbf{x_1}$, $\mathbf{x_2}$ and $\mathbf{x_3}$. Then we have $\mathbf{s} = [\mathbf{x_1^T}\ \mathbf{x_2^T}\ \mathbf{x_3^T}]$. The interaction matrix can now be obtained as $\mathbf{L_s} = [\mathbf{L_{x_1}}\ \mathbf{L_{x_2}}\ \mathbf{L_{x_3}}]^T$. Since, there are some configurations for which $\mathbf{L_s}$ is singular and four distinct camera poses where global minima exists, more than three points are usually considered in practice. If Z is known, we can use $\widehat{\mathbf{L_s^+}} = \mathbf{L_s^+}$. The other popular approach is to use $\widehat{\mathbf{L_s^+}} = \mathbf{L_{e^\star}^+}$, where $\mathbf{L_{e^\star}}$ is value of $\mathbf{L_s}$ at desired position. In this case $\widehat{\mathbf{L_s^+}}$ is constant, and only the desired depth of each point has to be set, which means no varying 3-D parameters have to be estimated during the visual servo. The calculation for interaction matrix using other features and in other coordinate system can be easily find in literature like [ECR92, SHV06, CH07, Cor11].

2.3.3 Hybrid Visual Servoing (HVS)

The advantages of both PBVS and IBVS are combined in hybrid approaches to visual servoing [CH07]. Among hybrid approaches, 2-1/2D visual servoing [MCB99] is well established from an analytical point of view. In 2-1/2D visual servoing, the control of camera rotational and translational degrees of freedom are decoupled i.e. control input is expressed in part in 3D metric/Euclidean space and in part in 2D image space. IBVS is used to control the camera translational degrees of freedom while PBVS is used to control rational degree of freedom. This approach provides several advantages. First, since camera rotation and translation controls are decoupled, the problem of camera retreat is solved. Second, 2-1/2D HVS is free of image singularities and local minima. Third, this method does not require full pose estimation, and rotation can be estimated without complete 3D estimation. Fourth, geometric camera motion and image plane trajectory can be controlled simultaneously. Finally, this method can accommodate large translational and rotational camera motions. However, there are a few problems associated with 2-1/2D HVS. One of the problems is the possibility of features leaving image boundaries. Next, it is still sensitive to noise and camera calibration errors. Finally, the selection of reference feature point affects the performance of 2-1/2D HVS.

In our work, we used motion control based on IBVS, which is discussed in Sect. 4.

Chapter 3

Vision-based Navigation of Mobile Robots

Visual information has already been widely used for mobile robot navigation. The increase in processing power, decreasing price of vision sensors, and ability to provide large amount of information from the scene with less power consumption have directly contributed to vision to become mainstream for the applications like mapping, localization, autonomous navigation, path following, etc. Traditionally, vision-based navigation solutions have mostly been devised for autonomous ground robots, but recently, visual navigation is gaining more and more popularity among researchers for flying robots or aerial robots and also for underwater robots. In this chapter, we present an overview of works in visual navigation especially for mobile robots. We begin the chapter with different types of navigation techniques based on the operating environment (Sect. 3.1), how robot knows the navigating environment (Sect. 3.2), and model of environment (Sect. 3.3). Then we talk about some mapping paradigms existing in literature in Sect. 3.4. Finally, we present and discuss some state of the art methods on appearance-based navigation for indoor mobile robots with particular focus on type of features used in Sect. 3.5. An exhaustive survey of Vision-based Navigation is available in [DK02, BFOO08, GFO15, FPRARM15].

3.1 Indoor and Outdoor Navigation

Depending upon the environment, the navigation can be divided into indoor and outdoor navigation.

3.1.1 Indoor Navigation

Indoor environments are mostly structural and human made. The indoor navigation can be mainly divided into: a) navigation in room, and b) navigation in corridors. For the former part, it is more likely to have abundant distinctive local features and global features whereas for the later part, surface may have similar texture or limited texture, reflections and repetitions, which make standard feature detection, matching/tracking to perform poorly. Also, robot has to cope with change in illumination as it travels between rooms and corridors. Despite these difficulties, mobile robots have been successfully navigated in indoor environment [BFOO08, GFO15]. However, there still lacks generic framework that works in most of indoor environments.

3.1.2 Outdoor Navigation

Outdoor navigation is also divided into two types a) Navigation in Structured Environments, and b) Navigation in Unstructured Environments. In former case, there are regular structures and features that can be tracked. The navigation is like some sort of road following [YO03]. This type of research is one of the hot topic in today's world especially for the self-driving cars or for providing driving assistant, where vision plays important role [CMM09, CMEM09, DSRC11]. In unstructured environment, there are no regular properties that can be tracked for navigation. In such cases, the robot randomly explores the vicinity or executes a mission with a goal position. One of such examples is Mars micro-rover [MGH^{+}95]. A common problem in all outdoor navigation systems is caused by variations in the illuminations resulting difference in contrast and textures of same scene taken at different times of day, different times of year, different weather conditions like fog, snow, sunshine, rain etc., which impose serious limitations on the performance. Some of the recent methods to cope this problems include [NSP13, GMMW10, DM13].

3.2 Map and Mapless Navigation

Depending upon navigation techniques, the overall navigation can be divided into a) Map-based Navigation, b) Map building-based Navigation, and c) Mapless navigation [DK02, BFOO08].

3.2.1 Map-Based Navigation

In map-based navigation, the environment in which robot has to navigate is known. In other words, the sequence of landmarks expected to find during the navigation is known and the task of vision system is to search and match the landmarks found in the on-line image with expected and known landmarks. When the landmarks are recognized, robot can use map to localize itself. This process is known as self-localization. However, for this type of navigation, the map should be built in advance. The main steps involved during navigation are as follows: [BEF96]

– Acquire Image.

– Detect landmarks (edges, corners, lines, blobs, colors, objects etc.).

– Match observed landmarks with expected ones that are stored in the map.

– Update the current position, i.e. localize in the map based on result of matching.

The localization in the map is further divided into a) Absolute localization or global localization, where the initial position of the robot is unknown at the beginning of the navigation, and b) Successive Localization or incremental localization, where the position of the robot is known or absolutely known at the beginning of the navigation. Depending upon the representation of the environment, navigation may be based on global coordinate system using metric maps [BD14, BSBB06] or in nodes based system using topological maps [VSSV00, RC07, BTZK07, CMM08] or mixture of two using hybrid maps [ŠRDC09, RLDL07].

3.2.2 Map building-based Navigation

In many real-world applications it is impossible to provide a map of the environment a-priori. In this map-building type of navigation, the robot can explore the environment and build its map by itself. One the earliest example was carried out by the Stanford CART robot [Mor90]. The most common methodologies based on this technique is Simultaneous Localization and Mapping (SLAM) or Concurrent Mapping and localization (CML) [GMC06, DRMS07, SRD06, ZK11, HS12, ESC14, MAMT15], which essentially perform three simultaneous tasks: navigation, mapping and localization. Although metric SLAM is the dominant theme in the literature, there also exists SLAM that does not use metric space and based upon appearance [MW10, GMMW10, CN11, MMW12] in which loop closing is performed in the topological space.

3.2.3 Mapless Navigation

This category includes all the navigation approaches that do not need and use any prior description of the environment. In the approaches discussed in this section, no maps are ever created. The robot navigates in the environment by detecting and extracting relevant information of observed elements of the environment like walls, doorways, desks, etc. The navigation decision is based on what robot perceives the environment. Such approaches mimic the navigation techniques used by animals especially insects. Some of the prominent approaches in this technique are presented below:

Navigation Using Optical Flow: These solutions estimate the motion of features within the sequence using optical flow. One of such examples is to calculate time-to-contact to avoid an obstacle [AC09], which is calculated on the basis of optical flow. In [SVSCG93], robot localizes itself using difference between the velocity of the image seen from the stereo pair. The robot will move so as to minimize the difference in the velocities.

Navigation based on Feature detection and/or Feature Tracking: These solutions use feature detection and/or matching/tracking to detect specific characteristics of the environment for the navigation. [FOPV13, PKB14, VSSV00] performed vision-based corridor navigation using the vanishing point extracted from corridor guidelines for the Nao humanoid robot and a wheelchair respectively. [LP02] uses two pairs of corresponding corner features to compute ground plane homography to determine the rotational motion of the robot.

3.3 Model-based and Appearance-based Navigation

The robotic system needs a representation of its environment to perform the navigation task from the initial point to a desired one. In order to perform this task autonomously, the environment should provide enough information for localizing initial and desired positions, defining path the robot has to follow between these two points, and controlling motion during navigation. The environment can be represented either in sensor space or in the 3D geometric space. There are generally two approaches that are widely used, which are briefly described below.

3.3.1 Model-based Approach

The first approach relies on the knowledge of an accurate and consistent 3D model of the navigation space, and the representation is geometric and environment centered. The landmark information is stored explicitly, and expressed in the coordinates of geometric frame [DRMS07, VLF04, RLDL07]. The navigation is then performed by matching the global model with a local model deduced from sensor data. Such a model can be computed from different features like lines, planes, or points [HZ03], or estimated from a learning step. Most of the simultaneous localization and mapping (SLAM) methods [DRMS07, GMC06, ZK11] fall in this category, but in this case autonomous motions are performed for discovering new areas rather than reaching desired position. The extensive review of SLAM is presented in [FPRARM15]. In the model-based approach, the environment is normally represented by metric maps, which will be discussed in Sect. 3.4.

3.3.2 Appearance-based Approach

The second approach, also known as appearance-based approach, does not require a 3D model of the environment, but it has instead the advantage of working directly in the sensor space. To simplify the process of appearance-based navigation, the navigation environment is generally represented by a graph [YITY05, GNTVG07, CMM08, DSRC11]. The nodes of the graph give characteristic features or zones of the environment (locations) obtained using the sensor data, which are mainly reference/key images, and arcs define the possibility for the robot to move between the two associated positions. Such maps are built in a prior offline mapping phase. Navigation is then usually performed by computing a similarity score between the view acquired by the camera and the different images of the database, or by using the features extracted from previous images via tracking and generating associated control command. This similarity can be based on global descriptors, like considering the whole image [Lab07, DM13], color histograms [WSM08], or image gradient [KZBD03]; or by using local descriptors, like photometric invariants [GNTVG07] or local feature points like corners, SIFT/SURF points or MSER [DSRC11, ŠRDC09, CMM08, BTZK07] or bag of words [CN11]. Recently, [MW10, GMMW10, CN11, MMW12] also performed SLAM using appearance only, which mainly focuses on building map and exploring unknown environment. More details of this approach are discussed in Sect. 3.5.

3.4 Metric Map-based Navigation and Topological Map-based Navigation

For mapped-based navigation, map has to be built in prior. Regarding robotic map building, metric, topological, and hybrid mappings are the paradigms that are generally accepted. Metric maps represent the environment in the geometric space, whereas topological maps describe the environment by help of connected graphs.

3.4.1 Metric Maps

Metric maps represent world as accurate as possible maintaining geometric details like distance, size etc. They are usually referenced according to a global coordinate system, which is most appropriate for localization, path planning, and obstacle avoidance. The recognition of places is based on geometry that is non-ambiguous and view point-independent. However, they are more difficult to make and maintain; they are space consuming and may result in inefficient planning as resolution does not depend upon the complexity of the environment; and they require accurate determination of the pose [Thr98]. An early representative of metric maps is occupancy grid mapping algorithm [Elf89, GFF08], which represents maps by fine-grained grids that model the occupied and free space of the environment. The other approach is landmark-based maps [SLL02, KM07], which identify and keep the 3D location of certain salient features in the environment. The representation of space through isolated landmarks reduces computational cost and memory requirements. However, these types of maps are not ideal for obstacle avoidance because lack of a landmark in a place does not imply that the space is free. Nevertheless, these representations are the most suitable when the determination of the pose is more important than the map.

3.4.2 Topological Maps

Topological maps [RC07, BTZK07, CMM08, GFO15] are the graphical representation of the environment in which nodes give characteristic feature or zone of the environment (location) and arcs give adjacency relations between locations. The key advantage over metric maps is their compactness, which permit fast planning, require less space, and provide natural interface to human such as go to kitchen etc. The resolution of topological maps corresponds directly to the complexity of the environment. They do not require exact determination of the geometric pose, which make them often better recover from drift and slippage that need to be constantly monitored and compensated in grid-based approach. The other advantage is that they are easily scalable. However, it is difficult to construct and maintain in large-scale environments if sensor information is ambiguous, and recognition of places is often difficult as they are sensitive to the point of view.

3.4.3 Hybrid Maps

Hybrid maps [RC07, TNS01, AFDM08, FM03] combine both such that they try to maximize the advantage and minimize the problems in metric and topological mapping techniques. Metric maps are generally finer grained than topological ones. So, in hybrid approach local metric maps are connected by global topological maps. In other words, the global environment is represented topologically, whereas the local environment is represented geometrically.

3.5 Appearance-based Navigation for Mobile Robots

In this section, we are going to discuss in detail about the appearance-based navigation approach, which is used in our work. As the appearance-based navigation is based on the pixel information of the image, following issues have to be taken into consideration:

- appropriate way of representing the environment;
- whether to use global features like color, histogram or local features like points;
- on-line similarity measurement which defines how the features are matched or (and) tracked between reference image and acquired image; and
- recording of images in the training phases as well as in subsequent image matching or (and) tracking process and how the reference images are selected.

To simplify the process of appearance-based navigation, the navigation environment is generally represented as topological rather than geometric. The selection of feature depends upon the environment conditions in which robot has to navigate. The similarity measure and selection of reference image depend upon feature selected. First, we discuss top level approach for appearance-based navigation based on key/reference images used to represent the environment. Then, we discuss state of the art methods especially focused on feature selection. Finally, we talk about navigation using visual memory and our approach for coming chapters.

3.5.1 Overview

Before navigation, the map of the environment has to be created. Navigation path is defined by series of key/reference images organized in an adjacency graph. The block diagram of general appearance-based navigation is shown in Fig. 3.1. (Note: This is not applicable to appearance-based SLAM methods and those that do not use topological graphs.)

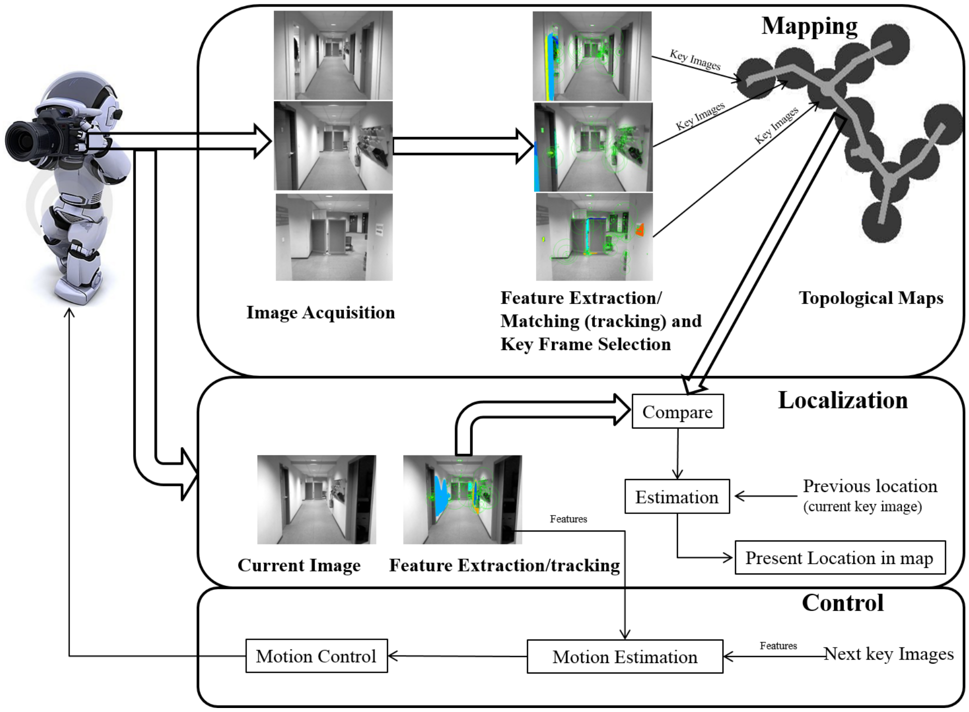


Figure 3.1 – Appearance-based Topological Navigation.

During the learning (mapping) phase, the features of the representative images (key/reference images) are captured from the environment and associated with the corresponding locations (nodes of graph). This set of selected key/reference images is termed as **image memory** or **image database** in the later sections. For navigation, robot first has to localize in the map. Localization in this case is to find two adjacent nodes of graph which match best with the current view. In the very first localization phase, the features of input image are compared with that of the reference images in the database. The location where the reference image best matches the input image gives the initial location of input image in the map. Therefore, the main goal is to find the images in the database that are similar in appearance to the input query image. After initial localization, the adjacency relation between nodes makes it possible to compare the features of the current image only with the images taken at currently believed locations and its immediate neighbor. This limits the number of the reference images used for examining, which generally results fast comparison process. For comparison of features, we can use either matching process or the knowledge from the previous image via feature tracking or both (if possible). The initial localization is purely based on a feature matching process whereas the later navigation process is generally based on feature tracking for efficiency. The pure matching process is employed in navigation in such cases whenever tracking of features are neither possible nor sufficient. Based on the previous location and current comparison result, the present location can be estimated. Based upon this present location and target location, the control command can be generated for visual navigation.

3.5.2 Methods Based on Global Descriptors

Global descriptors use entire image to describe it. They are normally fast to compute and economic to store in terms of memory requirements. Following are the navigation methods that uses global descriptors for appearance based topological navigation. [GWSV00] uses Principal Component Analysis to describe images in eigenspace. [UN00] uses six one-dimensional color histograms: three extracted from the HLS color space and other three extracted from the RGB color space with Jeffrey divergence used as a distance measure between two histograms, where they obtain accuracy of 87.5% in indoors. Three-dimensional histograms in RGB space (appearance) combined with odometric information and Bayes Filters have been used by [WSM08]. In [RMG14], Omni-Gist has been used in a semantic labeling process for building indoor topological maps. This place classification module is integrated with a Hidden Markov Model to ensure the temporal consistency. In [MZPI04], the image is represented by the first 15 Fourier coefficients (15 lowest frequency components), known as Fourier signatures. To overcome the perceptual aliasing problem, they fuse this image representation with a particle filter. In [ML04] omnidirectional images have been compared using the Manhattan distance function to determine both the translation and rotation angle for the visual homing. All of the above navigation system used omnidirectional camera. The work of [KZBD03] uses gradient orientation histograms as image descriptor and Euclidean distance to match histograms, and Learning Vector Quantization to find reference images. Multidimensional histograms are used to describe the global appearance features of an image such as colors, edge density, gradient magnitude, and textures by [ZWT03], where they achieved 83% accuracy in indoors. Whole Image SURF has been used by [XBH14] for outdoor localization. [WZG14] presented a loop closure detection method in which images are smoothed using a Gaussian kernel, and then resized to a small patch. The patch is then binarized to produce a binary code of a few hundred bits. The mutual information for the image pair is used as a similarity measure.

They are able to detect loop closures in a map of 20 million key locations. [DM13] have used Mutual Information for appearance-based visual path following which is somehow robust to illumination variations and seasonal change in outdoor. [MWP04] use color segmentation for their hybrid SLAM, known as RatSLAM, which mimics the hippocampal complex in rodents. [GMMW10] combine RatSLAM with Fab-Map in order to address the challenging problem of producing coherent maps across several times of the day.

Global descriptions work well for capturing the general structure of the scene, but they are not able to cope well with several visual problems like partial occlusions or camera rotation. Besides that, they have poor discriminating power and high perceptual aliasing effect.

3.5.3 Method Based on Local Descriptors

Local features are usually more robust to occlusions and changes in scale, rotation, translation and illumination. The recovery of relative poses between images, which can be used for confirming if two images come from the same scene, can be performed easily. However, the storage requirements and the computational cost are higher than for global descriptors. [KL04] use SIFT features and performed a global localization process based on a simple voting scheme. To cope with dynamic changes in the environment, they incorporate additional knowledge about neighborhood relationships between individual locations using a Hidden Markov Model. In [LK06], they presented a feature selection strategy in order to reduce the number of keypoints per location by measuring the discriminability of the individual features to describe each topological location. In order to handle both strong perceptual aliasing and dynamic changes of places efficiently, [KTTH11] has used Position-Invariant Robust Features (PIRFs) for the incremental appearance based SLAM. PIRFs are generated by averaging the SIFTs features, which appear to be slow-moving relative to the change in camera positions. These slow-moving features are identified using simple feature matching. A local feature that appears repeatedly in many sequential images is regarded as a slow-moving local feature. [RZL03] use Kanade-Lucas-Tomasi (KLT) tracker for matching persistent features in a sequence of omnidirectional images and constructed a topological map incrementally. [BTZK07] has used omnidirectional vision system (hyperbolic mirror + ordinary camera) and SIFT features for the appearance-based topological navigation. The heading estimation has been done by epipolar geometry assuming that the robot moves in a planar surface. [DSRC11, ŠRDC09] have demonstrated hybrid model for topological navigation based on a visual memory. Local 3D reconstruction has been used for verifying the key-point matches and automatic key frame selection using SIFT, Multi Scale Harris, and MSER features. The centroid of the features is used for motion control. This method is partially robust to occlusion and lighting variation and has been tested in real time in outdoor environment. MSER, SIFT and GLOH are used to create a signature of place by [RTA$^+$09] for localization purposes. They showed that these combinations increase notably the performance compared with the use of one descriptor alone. [BSC09] has developed omnidirectional vision and 2D laser range finder-based navigation system based on the Feature Stability Histogram and the textured-vertical edges.

3.5.4 Methods Based on Bag of Words (BoW)

Bag of Words are obtained by quantizing local features according to a set of feature models called visual dictionary, visual vocabulary, or codebook and then representing images as histograms of occurrences of each word in the image. BoW methods can efficiently index a huge amount of images incorporating a hierarchical scheme and an inverted index structure. However, the effect of perceptual aliasing worsens due to the quantization process, and there are no spatial relations between the words. [WCZ05, WZC06] present a global localization system based on BoW, where Harris Corners are described using the SIFT algorithm. An epipolar geometry step is incorporated in order to verify the loop candidate. [FEN07] apply hierarchical dictionary to the visual navigation problem, presenting a highly scalable vision-based localization and mapping method using image collections. Geometric verification with RANSAC is employed to determine if the image closes a loop or if a new place has to be added to the map. They use local geometric information to navigate within the generated topological map. Fast Appearance-Based Mapping (FAB-MAP) approach proposed by [CN11, CN08] under the assumption that modeling the probabilities that the visual words appear simultaneously can help in the localization process. [PN10] incorporates the spatial arrangement of the visual words to improve localization accuracy. Local features are quantized in both feature and image space and a set of statistics regarding their co-occurrence at different times of the day are calculated in [JY13] for appearance-based localization throughout the day. [LZLS13] proposes a place recognition system using MSLD to describe lines. A hierarchical visual dictionary was trained using these vectors, which was employed in combination with a Bayes filter for detecting loop closures in indoor environments.

3.5.5 Methods based on Combined Descriptors

There also exist various solutions for appearance-based navigation using different image descriptors in order to take benefits of each approach. A common approach is to use a global descriptor to perform a fast selection of searching candidate images and then use a more accurate process such as matching local features in order to confirm this association. [GNTVG07] has used rotation reduced and color enhanced SIFT (descriptors not invariant to rotation) and Invariant Column Segments (10 element vector) formed by 3 color invariants and 7 intensity invariants from Discrete Cosine Transformation of the gradient of the vertical lines as local feature and color as global feature for the omnidirectional vision-based topological navigation. [CMM09] has used the global descriptors computed by cubic interpolation of a triangular mesh and patch correlation (ZNCC) around Harris corners for visual memory-based navigation using generic camera model. [MSG$^+$07] proposes a three-step hierarchical localization method for omnidirectional images: set of candidate images selection using a global color descriptor, selection of correct reference images by matching line features described by their line support regions, and metric localization by 1D radial trifocal tensor. In [MGS07], their previous work is expanded to incorporate SURF features. [WY13] uses Orientation Adjacency Coherence Histograms for coarse localization and Harris Corners described by the SIFT descriptor verified by RANSAC + epipolar geometry for fine localization. [WTMZ07] uses similarities computed using two global descriptors: Weighted Gradient Oriented Histogram and Weighted Grid Integral Invariant to update the weights of a particle filter for outdoor localization. SIFT is used as an alternative to compute the position of the robot in those cases where it cannot be inferred using the combined global descriptors method. A location recognition system which combined edges, local features and color histograms has been proposed by [WY12].

3.5.6 Navigation using Visual Memory

The visual memory approach has been introduced in [MII96b] for conventional cameras and extended in [MIII99] for omnidirectional cameras. In [VSSV00], the vanishing point is used for the heading control whereas an appearance-based process is used to monitor the robot position along the path. A set of reference images are acquired manually at relevant positions along the path which correspond either to areas in the workspace where some special action can be undertaken (e.g. doors, elevators, corners, etc.) or viewpoints where very distinctive images can be acquired. During navigation, these reference images are compared with current images using the Sum of Squared Differences (SSD) metric. The position-based schemes relying on the visual memory approach have been proposed with 3D reconstruction using an EKF-based SLAM by [GNTVG07], or SfM by [RLDL07]. The work in [CMM08] has demonstrated indoor navigation of a mobile robot using a visual memory for both perspective and omni directional cameras. The robot is controlled by visual servoing based upon the regulation of successive homographies. The visual memory is selected from the tracking of points based on particle filter. [CMM09] have presented outdoor navigation based on decomposition of Essential matrix for generic cameras. In [CB09], a qualitative visual navigation scheme based on some heuristic rules is presented. [CC09] have used a time independent varying reference to cope with discontinuity in rotational velocities when key images are switched. In [ŠRDC09, DSRC11], the authors have demonstrated a hybrid model for topological navigation based on a visual memory in an outdoor environment. Local 3D reconstruction has been used for verifying the key-point matches and automatic key-frame selection using SIFT, Multi Scale Harris, and MSER features. Lucas-Kanade Differential Tracking with Isotropic Scaling and Contrast Compensation has been used for feature tracking over the views instead of matching. Baseline feature matching has been done only when tracking is not possible. However, the motion control was still based upon 2D features, in particular, the centroid of matched points. They also show that it is not necessary to converge towards each intermediate position (key frames) as long as it is possible to reach the final position. An image-based control scheme for driving wheeled mobile robots along visual paths has been proposed by [BSMH14], where the feedback of information given by geometric constraints: the epipolar geometry or the trifocal tensor without decomposing them into rotation and translation. Hence, the use of qualitative servoing [RMC06] eliminates the necessity of a database accurate enough to get satisfying trajectories regarding the initial and desired positions, contrary to [BTZK07] where the robot converges to the intermediary positions using visual servoing by minimizing the error between the current and successive desired positions of visual landmarks.

From the above literature, one can then conclude that accurate mapping and localization are not mandatory for visual navigation. In this respect, the goal of this thesis is to adopt this approach for indoor navigation. We propose a complete framework for selection of key/reference images to build a map, localization in the map, and motion control so that robot follows the visual route represented by its visual memory using entire image and local features. To our knowledge, the closest works to ours that use image memory are [DSRC11, ŠRDC09, CMM08], which, however, still use 3D information for navigation. The approach proposed in this thesis is instead different from the available literature as our method *only* exploits 2D information from the image without need of any 3D. First the mutual information is exploited for the navigation using entire image. Then, we use the same approach for indoor navigation based entirely on the line segments detected in the image. Finally, we combine line features with classical point features for more robust navigation in wide range of indoor scenarios.

Part II

Our Contribution

CHAPTER 4

Overview of the proposed method

THIS chapter gives the general overview of our method for mapping and navigation without considering collision avoidance. The presented approach follows the same proposed by [ŠRDC09], but for indoor environment with different features using 2D image information only. The proposed method of navigation is based on an image memory. Therefore, the image memory needs to be constructed in a prior. Hence, the overall process is divided into: a) off-line mapping (learning) phase, and b) on-line navigation phase.

Constraints

We consider a non-holonomic mobile robot of unicycle type equipped with a fixed perspective camera as the only sensing modality. The intrinsic parameters of the camera are constant and coarsely known. The presented framework (also in Chapters 5-7) is concerned only with a goal-directed behavior without considering obstacle avoidance. Thus, in the navigation experiments we assume that other moving objects will adopt collision-free trajectories, while a human supervisor is responsible for handling the emergency stop button. The devised control scheme exhibits a qualitative path following behavior, since the learned path in general is not tracked precisely. It is therefore suitable to prefer the center of the free space during the acquisition of the learning sequence. During navigation, it is assumed that the robot is initially inside the mapped environment. The localization outside the mapped location is out of scope of this work.

1) Mapping (Learning) Phase

During learning, the robot is moved in the navigation environment under the supervision of an operator capturing the images of the environment. From this learning sequence, a small subset of images is selected, which is known as image memory. This image memory consists of key/reference images that represent particular location of the environment. The selected key/reference images are organized in an adjacency graph, which gives the topological representation of the environment. The first image of the learning sequence is always selected as key/reference image. The other key images are automatically selected such that they are sufficiently different from the adjacent key frame. Such differences are determined from a similarity metrics, which depends upon the feature(s) selected. For the case of local geometric features, this similarity metrics is obtained from feature matching/-tracking. In other hand, for the global features before calculating similarity metrics, it is necessary to align the common portion of images (register images). The last image of the learning sequence is also stored in the database, which helps in determining when the robot has to stop at the end of the navigation. The general block diagram of mapping (learning) phase is shown in Fig. 4.1. The details of the selection of key images using different features are explained in their respective chapters.

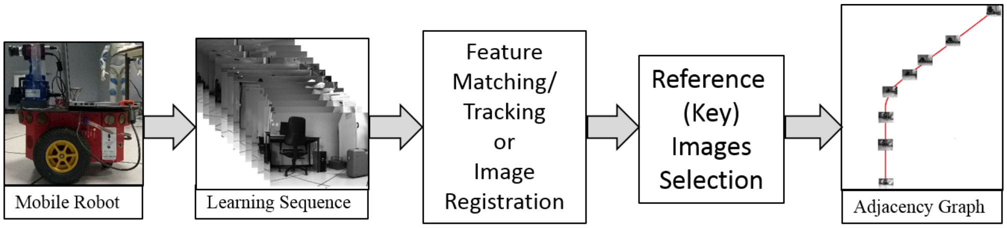

Mobile Robot
Learning Sequence
Feature Matching/ Tracking or Image Registration
Reference (Key) Images Selection
Adjacency Graph

Figure 4.1 – Mapping (Learning) Phase.

2) Navigation Phase

After the learning phase, the robot is ready to navigate autonomously in the mapped location. The navigation task is further divided into two subtasks: a) qualitative localization in the map and b) motion control. The overview of the proposed navigation framework is shown in Fig. 4.2.

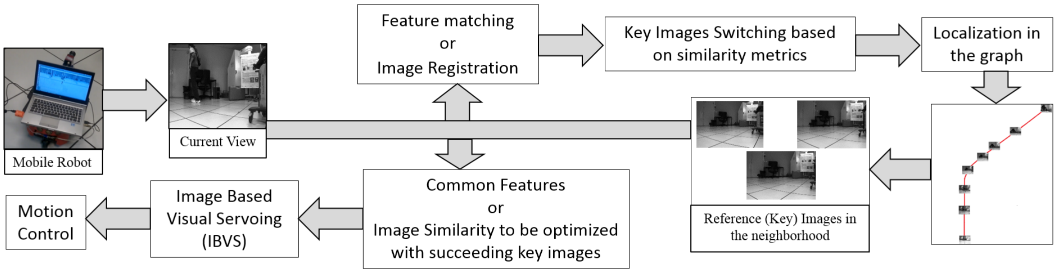

Feature matching or Image Registration
Key Images Switching based on similarity metrics
Localization in the graph
Mobile Robot
Current View
Reference (Key) Images in the neighborhood
Motion Control
Image Based Visual Servoing (IBVS)
Common Features or Image Similarity to be optimized with succeeding key images

Figure 4.2 – Navigation Phase.

a) Qualitative Localization in the map

In our case, localization in the map is qualitative rather than quantitative. The objective is to find the adjacent key images that matches the best with the current view. In Fig. 4.3, I_a is the current view of the robot. The position of I_a in the map is between key images I_P and I_N. The best matches are obtained either from feature matching or image registration.

For autonomous navigation, two types of localization have to be performed: a) initial localization, and b) successive localization.

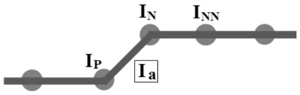


Figure 4.3 – Localization in map.

i) Initial Localization

Initial localization refers to localize the robot globally in the map, where the current view is compared with entire image memory to find the best adjacent key images. Navigation begins with the initial localization, which enables to start the robot from any position within the map. Initial localization can be also equally applicable to recover the robot if it is temporarily lost. If $\mathfrak{f}(...)$ is the similarity metrics, the initial position of I_a can be obtained from

$$I_{k_i} = \arg\max_{I_k} \{\mathfrak{f}(I_a, I_{k_1}), \mathfrak{f}(I_a, I_{k_2}), ..., \mathfrak{f}(I_a, I_{k_n})\} \quad (4.1)$$

$$I_{k_j} = \arg\max_{I_k} \{\mathfrak{f}(I_a, I_{k_{i-1}}), \mathfrak{f}(I_a, I_{k_{i+1}})\} \quad (4.2)$$

$$\begin{array}{ll} I_P = I_{k_j} \text{ and } I_N = I_{k_i} & \text{if } i > j \\ I_P = I_{k_i} \text{ and } I_N = I_{k_j} & \text{if } i < j \end{array} \quad (4.3)$$

ii) Successive Localization

After initial localization, further localizations can be performed by comparing only few adjacent key images in the neighborhood. This successive localization goes along with the motion control, which helps to select the appropriate key images during navigation for control. In our method, during successive localization, Current Image (I_a) is compared with three key images: Previous Key Image (I_P), Next Key Image (I_N), and Second Next Key Image (I_{NN}). Key Image switching occurs when I_N precedes I_a. In this case, I_N becomes new I_P and I_{NN} becomes new I_N. If $\mathfrak{F}(...)$ denotes three-view based similarity criterion and $\mathfrak{f}(...)$ denotes two-view based similarity criterion, key image switching is done when

$$\boxed{\mathfrak{F}(I_a, I_N, I_{NN}) > \mathfrak{F}(I_P, I_a, I_N) \text{ or } \mathfrak{f}(I_a, I_{NN}) > \mathfrak{f}(I_a, I_N). \qquad (4.4)}$$

The two view criterion is necessary to prevent from being lost when three view criterion is not possible to compute.

b) Motion Control

For navigation, the robot is not required to accurately reach each key image of the path, or to accurately follow the learned path. In practice, the exact motion of the robot should be controlled by an obstacle avoidance module [CC13]. Therefore, translational velocity is kept constant and reduced to a smaller value when turning. Such turnings are automatically detected by looking at the commanded rotational velocity. The rotational velocity is derived using key images and current image within an IBVS control law [CH06]. The error to be reduced in IBVS is obtained from the difference of feature positions or gradient of the global feature. The navigation is qualitative because the robot approximately follows the learned path and does not converge to all intermediate key frames. In many cases during navigation, the key image switching occurs before convergence. Our objective is to make the robot approximately follow the learned path. IBVS is able to keep the error under certain bounds.

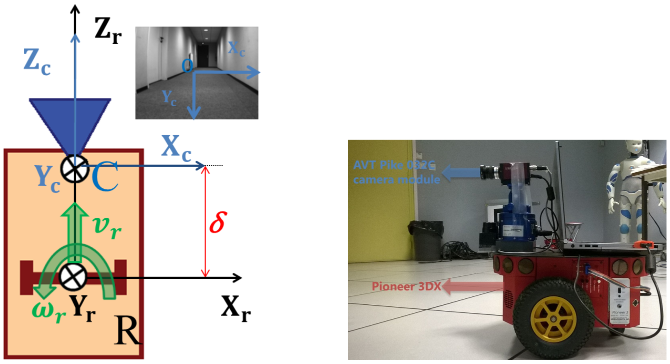


Figure 4.4 – Top view of robot (orange) equipped with a perspective camera (blue) with its optical axis perpendicular to axis of robot rotation (left) and experimental platform (right).

Let us define a vector of visual features as $\mathbf{s}$, the camera velocity expressed in camera frame as $\mathbf{u_c} = (v_{cx}, v_{cy}, v_{cz}, \omega_{cx}, \omega_{cy}, \omega_{cz})$ and the robot velocity as $\mathbf{u} = (v_r, \omega_r)$, where v is the linear velocity and ω is the rotational velocity around the given axes. The velocity of $\mathbf{s}$ can be related via an interaction matrix $\mathbf{J_s}$ [CH06] to $\mathbf{u_c}$ as

$$\dot{\mathbf{s}} = \mathbf{J_s}\mathbf{u_c}. \tag{4.5}$$

For the considered unicycle-like robot (Fig. 4.4 (left)), $\mathbf{u_c}$ can be expressed in terms of (v_r, ω_r) as

$$\mathbf{u_c} = \begin{pmatrix} -\delta\omega_r, & 0, & v_r, & 0, & -\omega_r, & 0 \end{pmatrix}, \tag{4.6}$$

where δ is the distance between the camera center and the robot center of rotation. From (4.5) and (4.6), we obtain

$$\dot{\mathbf{s}} = \mathbf{J}_v v_r + \mathbf{J}_\omega \omega_r, \tag{4.7}$$

where $\mathbf{J}_v$ and $\mathbf{J}_\omega$ are the Jacobian associated with v_r and ω_r respectively. In order to drive $\mathbf{s}$ to its desired value $\mathbf{s}^*$, we set the error $\mathbf{e}$ as

$$\mathbf{e} = \mathbf{s} - \mathbf{s}^\star. \tag{4.8}$$

The relationship between robot velocity and time variation of the error $\dot{e}$ can be obtained from (4.8) and (4.7) as

$$\dot{\mathbf{e}} = \mathbf{J}_v v_r + \mathbf{J}_\omega \omega_r. \tag{4.9}$$

In order to ensure an exponential decoupled decrease of the error i.e. $\dot{\mathbf{e}} = -\lambda\mathbf{e}$, we control ω_r such that from (4.8) and (4.9) we obtain

$$\omega_r = -\mathbf{J}_\omega^+(\lambda(\mathbf{s} - \mathbf{s}^\star) + \mathbf{J}_v v_r), \tag{4.10}$$

where λ is a positive gain, and $\mathbf{J}_\omega^+$ is the pseudo-inverse of $\mathbf{J}_\omega$. In practice, $\mathbf{J}_\omega$ and $\mathbf{J_v}$ have to be estimated or approximated because of involvement of unknown depth [CH06]. Since, we are controlling only one degree of freedom, one feature is sufficient to calculate ω_r. The calculations of $\mathbf{J}_v$ and $\mathbf{J}_\omega$ for different features are explained in their respective chapters.

Stability Analysis

In order to assess the stability of the closed-loop visual servo systems, we will use Lyapunov analysis. We consider the candidate Lyapunov function defined by the squared error norm i.e.

$$\mathscr{L} = \frac{1}{2}\|e(t)\|^2,$$

whose derivative is given by

$$\dot{\mathscr{L}} = \mathbf{e}^\mathbf{T}\dot{\mathbf{e}}. \tag{4.11}$$

From (4.9) and (4.11), we have

$$\dot{\mathscr{L}} = \mathbf{e}^T(\mathbf{J}_v v_r + \mathbf{J}_\omega \omega_r). \tag{4.12}$$

From (5), (4.8) and (4.12), we have

$$\dot{\mathscr{L}} = \mathbf{e}^T(\mathbf{J}_v v_r - \mathbf{J}_\omega \mathbf{J}_\omega^+(\lambda \mathbf{e} + \mathbf{J}_v v_r)). \tag{4.13}$$

For a unicycle robot, where the curvature is not bounded, [CCO08] shows that asymptotic stability is guaranteed using two features.

Using only one feature (which is sufficient to calculate ω_r), the terms $\mathbf{e}$, $\mathbf{J}_v$, $\mathbf{J}_\omega$, and $\mathbf{J}_\omega^+$ in (4.13) are simply scalar. For ideal case, in this situation $J_\omega^+ = \frac{1}{J_\omega}$. On simplification, (4.13) reduces to

$$\dot{\mathscr{L}} = -\lambda e^2 < 0 \quad \forall e \neq 0. \tag{4.14}$$

(4.14) ensures global asymptotic stability.

In practice, $\mathbf{J}_\omega$ and $\mathbf{J_v}$ have to estimated or approximated. Let they are represented by $\hat{\mathbf{J}}_\omega$ and $\hat{\mathbf{J}}_\mathbf{v}$. Let $\hat{\mathbf{J}}_\omega^+$ is the pseudo-inverse of $\hat{\mathbf{J}}_\omega$, (5) can be written as

$$\omega_r = -\hat{\mathbf{J}}_\omega^+(\lambda \mathbf{e} + \hat{\mathbf{J}}_v v_r), \tag{4.15}$$

From (4.15), and (4.12), we have

$$\dot{\mathscr{L}} = \mathbf{e}^T(\mathbf{J}_v v_r - \mathbf{J}_\omega \hat{\mathbf{J}}_\omega^+(\lambda \mathbf{e} + \hat{\mathbf{J}}_v v_r)). \tag{4.16}$$

On simplification of (4.16), we have

$$\dot{\mathscr{L}} = -\lambda \mathbf{e}^T \mathbf{J}_\omega \hat{\mathbf{J}}_\omega^+ \mathbf{e} - \mathbf{e}^T(\mathbf{J}_\omega \hat{\mathbf{J}}_\omega^+ \hat{\mathbf{J}}_v - \mathbf{J}_v) v_r. \tag{4.17}$$

In order to ensure asymptotic stability, $\dot{\mathscr{L}} < 0$. On simplification (4.17), we obtain following sufficient condition

$$\frac{\mathbf{e}^T(\mathbf{J}_v - \mathbf{J}_\omega \hat{\mathbf{J}}_\omega^+ \hat{\mathbf{J}}_v)}{\mathbf{e}^T \mathbf{J}_\omega \hat{\mathbf{J}}_\omega^+ \mathbf{e}} < \frac{\lambda}{v_r}, \tag{4.18}$$

under assumption $\mathbf{e} \notin \ker \hat{\mathbf{J}}_\omega^+$.

In (4.17), the first term is always negative if $\mathbf{J}_\omega \hat{\mathbf{J}}_\omega^+ > 0$. In second term, $\mathbf{J}_v$ (hence $\hat{\mathbf{J}}_v$), which consists of depth term, can be approximated to zero (we will show this in coming sections). Therefore, in this case $\dot{\mathscr{L}} < 0$. Using only one feature, the global asymptotic stability is assured when

$$\frac{J_\omega}{\hat{J}_\omega} > 0 \text{ and } J_v \simeq \hat{J}_v \simeq 0. \tag{4.19}$$

Experimental Setup

The experiments were performed with a Pioneer 3DX equipped with an AVT Pike 032C camera module as shown in Fig. 4.4. All computations, except for the low-level control, were performed on a laptop with 3-GHz Intel Core i7-3540M CPU. The image resolution in the experiments was 640×480. The mapping was done offline, whereas the navigation experiment was performed online in real-time. The acquisition of images and the high-level motion control for the Pioneer were done through the interface provided by ViSP [MSC05]. The image coordinates have been normalized by the camera intrinsic parameters before deriving the rotational velocity. The experiments have been performed in an indoor environment, i.e., inside a room and a corridor. Even though simple navigation path with linear and curved trajectories have been used in the experiments, the method can be easily extended for graphs with intersections and multiple paths. The trajectories have been obtained from the odometry of the Pioneer. Since our navigation is qualitative rather than quantitative, the odometry obtained from Pioneer 3-DX is highly accurate enough to serve our purpose.

CHAPTER 5

Mutual Information based Navigation

THIS chapter presents a complete framework for image-based navigation from an image memory that exploits mutual information and does not need any feature extraction, matching or any 3D information. As explained before, the path to be followed is represented by a set of automatically selected key images. The shared information (entropy) between the current acquired image and nearby key images is exploited to switch key images during navigation. Based on the key images and the current image following Sect. 4, the control law proposed by [DM13] is used to compute the rotational velocity of a mobile robot during its qualitative visual navigation. Using our approach, real-time navigation has been performed inside a corridor and inside a room with a Pioneer 3DX equipped with an on-board perspective camera without the need of accurate mapping and localization.

The work described in this chapter is presented in [BRGC16b].

5.1 Navigation using Entire Image

Robust extraction and matching/tracking of the features over a large environment is still a bottleneck for visual navigation schemes. However, another possibility is to directly exploit a comparison between the full key images and current image, using Fourier space [ZK09], cross-correlation [CMM09, MII96a], image intensity [CMC08, CM11], or mutual information [DM13]. In [VSSV00], Sum of Squared Difference (SSD) have been used for changing key images but the control law is based on the vanishing point of the corridor and the key images selection was not automatic. The work of [DM13] shows visual path following using mutual information in outdoor environment but it lacks an appropriate selection and switching of the key images. Also, the robot should be near to the starting point in the learning. Compared to intensity-based similarity metrics, like SSD or cross-correlation, mutual information is robust to illumination variations and to large occlusions [DM12]. Additionally, mutual information is a classic similarity measure especially for multi-modal registration techniques in medical imaging and remote sensing [PMV03, MHV$^+$01, VWI97]. Therefore, in this work we choose to use mutual information for navigation of a mobile robot in an indoor environment.

Our Contribution

In our work, we propose a complete method for the indoor navigation (mapping, localization in the topological graph and motion control) based on image memory using mutual information. We have extended the work of [DM13] with automatic key images selection, initial localization in the map, key images switching for successive localization and use of multiple key images for control. Indeed in [DM13], all the images of the sequence acquired during the learning step were used, and the synchronization between the current image and the corresponding image in the sequence learned was a real challenge as adequate switching of key frames are not employed. The use of few key images, initial localization and automatic switching of key images removes the problem of synchronization and enables the robot to use different forward velocities than those used for the learning phase. The initial localization makes it possible for the robot to start from any position within the map. Without the need of any accurate mapping and localization, 3D information, and feature extraction and matching, our method is able to perform navigation in indoor environment using mutual information.

5.2 Our Method

5.2.1 Mutual Information

In our case, mutual information is the information shared by two images. For two images I and I_k, mutual information is given by the following equation [VWI97]

$$MI(I,I_k) = H(I) + H(I_k) - H(I,I_k), \tag{5.1}$$

where $H(I)$ denotes the entropy of the image I, i.e., its variability. $H(I,I_k)$ denotes the joint entropy of the images I and I_k, i.e., the joint variability of two images. By subtracting the joint variability from the variabilities, as in (5.1), we obtain the shared information of the two images, which is defined as mutual information. The entropy of the image can be easily obtained from its probability distribution, i.e., the normalized histogram of the image. Let n be the total number of gray levels and $p_I(i)$ be the probability of the particular gray level i. The entropy of the image I is then defined as

$$H(I) = -\sum_{i=0}^{n-1} p_I(i)\log(p_I(i)). \tag{5.2}$$

Similarly, the joint entropy between I and I_k can be obtained from joint probability distribution, i.e., normalized joint histogram between two images

$$H(I,I_k) = -\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} p_{II_k}(i,j)\log(p_{II_k}(i,j)), \tag{5.3}$$

where $p_{II_k}(i,j)$ is the joint probability between gray level i in I and gray level j in I_k.

Equations (5.1-5.3) give the following expression for the mutual information

$$MI(I,I_k) = -\sum_{i,j} p_{II_k}(i,j)\log\left(\frac{p_{II_k}(i,j)}{p_I(j)p_{I_k}(j)}\right). \tag{5.4}$$

The analytical functions for the probabilities are given by [DM13]

$$p_I(i) = \frac{1}{N_x}\sum_x \phi(i - I(x)) \text{ and} \tag{5.5}$$

$$p_{II_k}(i,j) = \frac{1}{N_x}\sum_x \phi(i - I(x))\phi(j - I_k(x)), \tag{5.6}$$

where N_x is the total number of pixels in the image and $\phi(x)$ is a function used to fill the histogram such that $\phi(\xi) = 1$ if $\xi = 0$ and 0 otherwise.

For gray images, $n = 256$. If larger number of gray level are used for the construction of the histogram, there will be more empty bins. Moreover, the histograms with larger number of bins are expensive to construct in terms of memory as well as time. So, in practice, smaller value of n (like 8, 16) are used. The other advantage of having smaller number of bins is that the cost function is smoother with a larger convergence domain [DM13]. For $n = N_c$ bins, the scaled image $\overline{I(x)}$ is given as

$$\bar{I}(x) = I(x)\frac{N_c - 1}{255}, \tag{5.7}$$

which has no longer integer values of intensities. Thus, function ϕ in (5.5-5.6) has to be modified to use real values. The solution for this is to use a B-spline function (that corresponds to B-spline interpolation) [MCV$^+$97].

The similarity measure given by mutual information is meaningful if the shared portions of the images are aligned while calculating the joint probability. Therefore, to measure the similarity between two images, the images are registered with each other such that mutual information between them is maximized. The maximization problem is given as

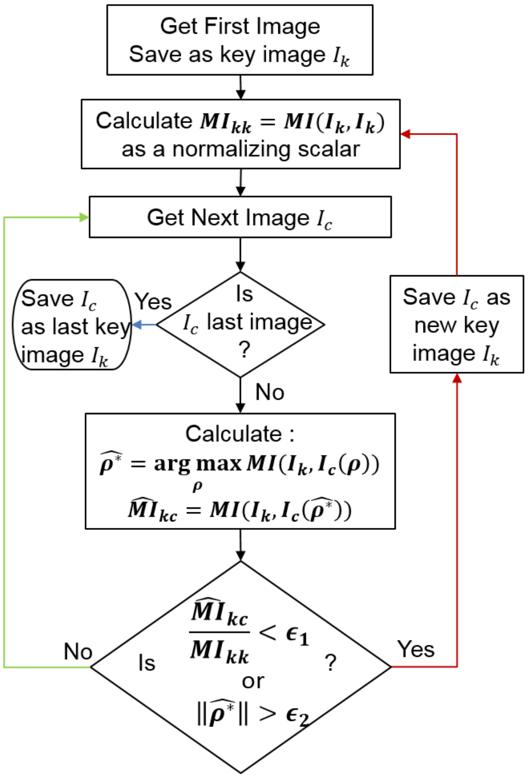
$$\hat{\rho} = \arg\max_{\rho} MI(I(\rho), I_k), \tag{5.8}$$

where ρ is the spatial transformation that maximizes mutual information. The mutual information in the optimal transformation $\hat{\rho}$ is given as

$$\widehat{MI}(I, I_k) = MI(I(\hat{\rho}), I_k). \tag{5.9}$$

In our approach, mutual information is calculated from (5.8-5.9) using the second order optimization of mutual information proposed by [DM12] with 8 histogram bins. Multi-scale registration [PMV98] has been performed instead of single scale, which enables to have fast and more accurate registration. Only 2D translation has been used as the transformation function during the registration in our experiments. In other words, the output of equation (5.8) is a 2D translation between the images. The choice of this transformation is reasonable as we are just controlling the vertical rotational velocity that defines the heading angle of the robot.

5.2.2 Mapping (Key Images Selection)



Get First Image
Save as key image I_k
Calculate $MI_{kk} = MI(I_k, I_k)$
as a normalizing scalar
Get Next Image I_c
Save I_c as last key image I_k
Yes
Is I_c last image ?
No
Save I_c as new key image I_k
Calculate :
$\widehat{\rho}^* = \arg\max_{\rho} MI(I_k, I_c(\rho))$
$\widehat{MI}_{kc} = MI(I_k, I_c(\widehat{\rho}^*))$
No
Is $\frac{\widehat{MI}_{kc}}{MI_{kk}} < \epsilon_1$ or $\|\widehat{\rho}^*\| > \epsilon_2$?
Yes

Figure 5.1 – Key Images selection using mutual information.

As already explained, in the proposed method, the environment is represented by a few number of key images that are selected automatically, unlike [DM13]. The key images selection procedure is sketched in Fig. 5.1. Let I_c be the most recently acquired image and I_k be the most recent key image. The new key image is selected if the mutual information metrics is smaller than a certain threshold, or the transformation is too large. The first criterion ensures that newly selected key frame is sufficiently different from the previous key frame, whereas the second criterion is essentially required to get a sufficient number of key images while turning.

5.2.3 Navigation in the Map

5.2.3.1 Qualitative Localization

Initial Localization in the map The navigation starts with the initial localization where the first acquired image (I_a) is compared with all the key images based upon the maximization of the mutual information (5.8-5.9). The key image that has maximum mutual information with I_a is selected. Let it be I_{k_i}. Then the adjacent key image with second maximum value of mutual information is also selected. This image can be either $I_{k_{i+1}}$ or $I_{k_{i-1}}$. If the robot is assumed to move along the same direction as the images arranged in the database, I_a is between $I_{k_{i-1}}$ and I_k, or I_k and $I_{k_{i+1}}$. For simplicity, we denote the previous key image as I_P and the next key image as I_N. If $\widehat{MI}(...)$ denotes the mutual information between images at optimal transformation, the initial localization can be expressed as

$$\boxed{\begin{aligned} I_{k_i} &= \arg\max_{I_k}\{\widehat{MI}(I_a,I_{k_1}),\widehat{MI}(I_a,I_{k_2}),......,\widehat{MI}(I_a,I_{k_n})\} \\ I_{k_j} &= \arg\max_{I_k}\{\widehat{MI}(I_a,I_{k_{i-1}}),\widehat{MI}(I_a,I_{k_{i+1}})\} \\ & I_P = I_{k_j} \text{ and } \quad I_N = I_{k_i}\text{if } i > j \\ & I_P = I_{k_i} \text{ and } \quad I_N = I_{k_j}\text{if } i < j \end{aligned}}.$$

Successive Localization (Key Images Switching) After the initial localization in the map, further localizations can be done by just comparing the current image with few adjacent key images. The next key image I_N and the second next key image I_{NN} are compared with the current image I_a. Then switching of key images is done when at least one of the following criteria is fulfilled for consecutive acquired images

$$\boxed{\begin{array}{ccc} \widehat{MI}(I_a,I_{NN}) & > & \widehat{MI}(I_a,I_N) \text{ or} \\ \dfrac{\widehat{MI}(I_a,I_{NN})}{MI(I_a,I_a)} < \varepsilon & \&\& & \dfrac{\widehat{MI}(I_a,I_N)}{MI(I_a,I_a)} < \varepsilon, \end{array}}$$

where ε is a small constant (say 0.15). The denominator in the second criteria is a normalization that helps to select ε for a given environment. The second criterion is essentially useful when the registration error is large or the robot is lost, which may sometimes occur with regions with no significant texture. If the error is still large in such case after switching, it is better to go back to initial localization. After switching the images, I_N becomes I_P, I_{NN} becomes I_N, and next key image from I_N becomes I_{NN}. Then the process repeats. When the end of the database is reached, I_{NN} will not be available and I_N will be the last image acquired during the mapping. Therefore, the navigation needs to be stopped. Otherwise, the robot will be moving out of the mapped environment. The coarse registration (with few iterations) using (5.8-5.9) is found to be sufficient for key images switching from our experiments.

5.2.3.2 Motion Control

During visual navigation, the robot moves so as to maximize mutual information with the current reference image. Comparing with classical visual servoing, the gradient of mutual information ($\mathbf{L_{MI}}$) gives the error term whereas the Hessian of mutual information ($\mathbf{H_{MI}}$) gives the interaction matrix $\mathbf{J_s}$. The derivatives of mutual information can be calculated by following [DM13]

$$\mathbf{L_{MI}} = \sum_k \mathbf{L_{p_{II_k}}} \left(1+\log\left(\frac{p_{II_k}}{p_{I_k}}\right)\right), \tag{5.10}$$

$$\begin{aligned} \mathbf{H_{MI}} \quad = \quad & \sum_k \mathbf{L}^T_{\mathbf{p_{II_k}}} \mathbf{L_{p_{II_k}}} \left(\tfrac{1}{p_{II_k}} - \tfrac{1}{p_{I_k}}\right) \\ & + \mathbf{H_{p_{II_k}}} \left(1+\log\left(\tfrac{p_{II_k}}{p_{I_k}}\right)\right). \end{aligned} \tag{5.11}$$

If N_x is the number of pixels considered in the images and ϕ is a twice differentiable B-spline function, the derivatives of the joint probability using $Nc\,(< 255)$ grey levels are as follows

$$p_{II_k}(i,j) = \frac{1}{N_x} \sum_x \phi(i - \bar{\mathbf{I}}(\mathbf{x}))\phi(j - \bar{\mathbf{I}}_\mathbf{k}(\mathbf{x})), \tag{5.12}$$

$$\mathbf{L_{p_{II_k}}}(i,j) = \frac{1}{N_x} \sum_x \mathbf{L}_{\phi(i-\bar{\mathbf{I}}(\mathbf{x}))}\phi(j - \bar{\mathbf{I}}_\mathbf{k}(\mathbf{x})), \tag{5.13}$$

$$\mathbf{H_{p_{II_k}}}(i,j) = \frac{1}{N_x} \sum_x \mathbf{H}_{\phi(i-\bar{\mathbf{I}}(\mathbf{x}))}\phi(j - \bar{\mathbf{I}}_\mathbf{k}(\mathbf{x})), \tag{5.14}$$

where,

$$\mathbf{L}_{\phi(i-\bar{\mathbf{I}}(\mathbf{x}))} = -\frac{\partial \phi}{\partial i}(\nabla\bar{\mathbf{I}}\mathbf{L_x}), \tag{5.15}$$

$$\mathbf{H}_{\phi(i-\bar{\mathbf{I}}(\mathbf{x}))} = \frac{\partial^2 \phi}{\partial i^2}(\nabla\bar{\mathbf{I}}\mathbf{L_x})^T(\nabla\bar{\mathbf{I}}\mathbf{L_x}) - \frac{\partial \phi}{\partial i}(\mathbf{L_x^T}\nabla^2\bar{\mathbf{I}}\mathbf{L_x} + \nabla\bar{\mathbf{I}}\mathbf{H_x}), \tag{5.16}$$

where $\nabla\bar{\mathbf{I}} = \begin{bmatrix} \nabla_x\bar{\mathbf{I}} & \nabla_y\bar{\mathbf{I}} \end{bmatrix}$ are the image gradients,

$\nabla^2\bar{\mathbf{I}} = \begin{bmatrix} \nabla_{xx}\bar{\mathbf{I}} & \nabla_{xy}\bar{\mathbf{I}} \\ \nabla_{yx}\bar{\mathbf{I}} & \nabla_{yy}\bar{\mathbf{I}} \end{bmatrix}$ are the gradients of image gradient, $\mathbf{L_x}$ is the interaction matrix of a point that links its displacement in the image plane to the camera velocity (refer (2.53)), and $\mathbf{H_x}$ is the Hessian of the point w.r.t. the camera velocity. The complete expression for $\mathbf{H_x}$ for six degrees of freedom is given in [LM04].

In order to maximize mutual information, we set v_r as constant and control ω_r. From (4.10), we get

$$\omega_r = -\mathbf{J}_\omega^+(\lambda(\mathbf{s}-\mathbf{s}^*)+\mathbf{J}_v v_r), \tag{5.17}$$

where λ is a positive gain, $\mathbf{J}_\omega^+$ is the pseudo-inverse of $\mathbf{J}_\omega$ and $(\mathbf{s}-\mathbf{s}^*)$ is the error term. $\mathbf{J}_v$ and $\mathbf{J}_\omega$ are the Hessian of mutual information w.r.t v_r and ω_r respectively whereas $(\mathbf{s}-\mathbf{s}^*)$ is the gradient of the mutual information. Since we are controlling only ω_r, in (5.17),

$$\begin{aligned} (s-s^*) = L_{MI\omega}, \quad & J_v = \tfrac{\partial L_{MI\omega}}{\partial v_r} = H_{MIv}, \\ \text{and} \quad & J_\omega = \tfrac{\partial L_{MI\omega}}{\partial \omega_r} = H_{MI\omega}, \end{aligned} \tag{5.18}$$

where $L_{MI\omega}$ is the Jacobian of mutual information w.r.t. ω_r, and H_{MIv} and $H_{MI\omega}$ are the Hessian of the mutual information w.r.t. v_r and ω_r respectively. They can be derived as follows:

Using (4.6), and the interaction matrix of point given in (2.53), we obtain\hat{L}_{P_{II_{k}}}

$$\mathbf{L}_{\mathbf{xu}\omega} = \begin{bmatrix} -\left(\frac{\delta}{Z}+1+x^2\right) \\ -xy \end{bmatrix} \simeq \begin{bmatrix} -(1+x^2) \\ -xy \end{bmatrix} \text{ for } \delta \ll Z, \tag{5.19}$$

where $\mathbf{L}_{\mathbf{xu}\omega}$ is the interaction matrix of the point that links its displacement in the image plane to ω_r, and δ can be safely neglected with respect to Z (depth of point from image plane). Hence, from (5.10-5.15) using (5.19) for $\mathbf{L_x}$, we can obtain $L_{MI\omega}$, which is a scalar. Differentiating (5.19) w.r.t. v_r and ω_r, we obtain $\mathbf{H_{xuv}}$ and $\mathbf{H}_{\mathbf{xu}\omega}$, which are the Hessian matrix of the point that links its displacement in the image plane to v_r and ω_r respectively. Therefore, we obtain

$$\mathbf{H}_{\mathbf{xu}\omega} = \begin{bmatrix} 2x(1+x^2) \\ y(1+2x^2) \end{bmatrix} \text{ and } \mathbf{H_{xuv}} = \begin{bmatrix} -\frac{2x^2}{Z} \\ -\frac{2xy}{Z} \end{bmatrix} \simeq \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{5.20}$$

since x^2 and xy can be safely neglected w.r.t. Z, where (x,y) is the normalized point in the image plane. From (5.10-5.16) using (5.19-5.20) for $\mathbf{L_x}$ and $\mathbf{H_x}$ respectively in (5.15-5.16), we obtain H_{MIv} and $H_{MI\omega}$, which are scalar. As $H_{MIv} \simeq 0$, (5.18) becomes

$$(s-s^*) = L_{MI\omega}, J_v \simeq 0, \text{ and } J_\omega = H_{MI\omega}. \tag{5.21}$$

Since visual servoing is known to be robust against modeling errors [CH06], such approximations in (5.19-5.21) are reasonable. Thus, from (5.17) and (5.21), the final expression for ω_r is obtained as

$$\omega_r = -\frac{\lambda L_{MI\omega}}{H_{MI\omega}}, \tag{5.22}$$

which is similar to the control scheme proposed in [DM13]. In order to smooth the rapid steering actions when switching between frames, a feed-forward command is also added to ω_r. The final expression for the rotational velocity is calculated by using I_N and I_{NN} as follows:

$$\omega_r = -\lambda\left(h_1\frac{L_{MI\omega(I_a,I_N)}}{H_{MI\omega(I_a,I_N)}} + h_2\frac{L_{MI\omega(I_a,I_{NN})}}{H_{MI\omega(I_a,I_{NN})}}\right), \tag{5.23}$$

where h_1 and h_2 are positive weights such that $h_1 + h_2 = 1$, $L_{MI\omega(I_a,I_N)}$ and $H_{MI\omega(I_a,I_N)}$ are derivatives of mutual information between I_a and I_N, and $L_{MI\omega(I_a,I_{NN})}$ and $H_{MI\omega(I_a,I_{NN})}$ are derivatives of mutual information between I_a and I_{NN}.

Thus, our complete framework uses only the information directly obtained from the images without any feature extraction or 3D information. From this information, we derive the required rotational velocity using *IBVS*, which makes the robot to follow the learned path successfully without any need for accurate mapping or localization.

Stability Analysis

If J_ω is the true Jacobian and $\hat{J}_\omega$ is the approximated Jacobian related to ω_r, we have

$$J_\omega = H_{MI\omega} \text{ and } \hat{J}_\omega = \hat{H}_{MI\omega}, \tag{5.24}$$

where

$$\begin{aligned} H_{MI\omega} &= \sum_k L^2_{P_{II_k}} \left(\frac{1}{p_{II_k}} - \frac{1}{p_{I_k}} \right) + H_{P_{II_k}} \left(1 + \log \left(\frac{p_{II_k}}{p_{I_k}} \right) \right) \text{ and} \\ \hat{H}_{MI\omega} &= \sum_k \hat{L}^2_{P_{II_k}} \left(\frac{1}{p_{II_k}} - \frac{1}{p_{I_k}} \right) + \hat{H}_{P_{II_k}} \left(1 + \log \left(\frac{p_{II_k}}{p_{I_k}} \right) \right). \end{aligned} \tag{5.25}$$

In (5.25), $L_{P_{II_k}}$ and $H_{P_{II_k}}$ has been derived using $\mathbf{L_{xu\omega}}$ whereas $\hat{L}_{P_{II_k}}$ and $\hat{H}_{P_{II_k}}$ has been derived using $\hat{\mathbf{L}}_{\mathbf{xu}\omega}$ similarly from (5.11)-(5.16) with

$$\mathbf{L_{xu\omega}} = \begin{bmatrix} -\left(\frac{\delta}{Z} + 1 + x^2 \right) \\ -xy \end{bmatrix} \text{ and } \hat{\mathbf{L}}_{\mathbf{xu}\omega} \simeq \begin{bmatrix} -(1 + x^2) \\ -xy \end{bmatrix}. \tag{5.26}$$

Since $\left(\frac{1}{p_{II_k}} - \frac{1}{p_{I_k}} \right) > 0$ and $\left(1 + \log \left(\frac{p_{II_k}}{p_{I_k}} \right) \right) > 0$ in (5.25) and we assume $\delta \ll Z$ in (5.26), from (5.24)-(5.26) we have,

$$\frac{J_\omega}{\hat{J}_\omega} = \frac{H_{MI\omega}}{\hat{H}_{MI\omega}} > 0. \tag{5.27}$$

From (5.21), we have Jacobian related to v_r is close to zero. Hence, with (5.27), the global asymptotic stability is guaranteed as inferred by (4.19).

5.3 Results and Discussions

The image resolution in the experiments was 640×480. Eight histogram bins and fourth order B-spline functions have been used for experiments. The experiments have been performed in an indoor environment, i.e., inside a room and a corridor with $\lambda = 1.0$, $h_1 = 0.7$, and $h_2 = 0.3$ in (5.23). Even though the proposed method has been validated via a simple navigation path with linear and curved trajectories, the method can be easily extended for graphs with intersections and multiple paths. The trajectories presented below are obtained from the odometry of the Pioneer Robot. The videos of the experiments are available in [1].

5.3.1 Mapping

1278 images have been acquired as learning sequence inside a robotics room. 10 images shown in Fig. 5.2 have been selected automatically from the mapping procedure described in Sect. 5.2.2 as key frames. Similarly, 36 images as shown in Fig. 5.3 have been selected as key images from 3881 images acquired for the 20m path consisting of the robotics room and a corridor. The trajectories obtained from the odometry are shown by a red curve in Figs. 5.4-5.7, where the red symbol $*$ represents the location of the key images. The obtained key images are able to represent the learned path. There are more key images over a small distance in case of turnings and of locations where the surface does not have good textures.

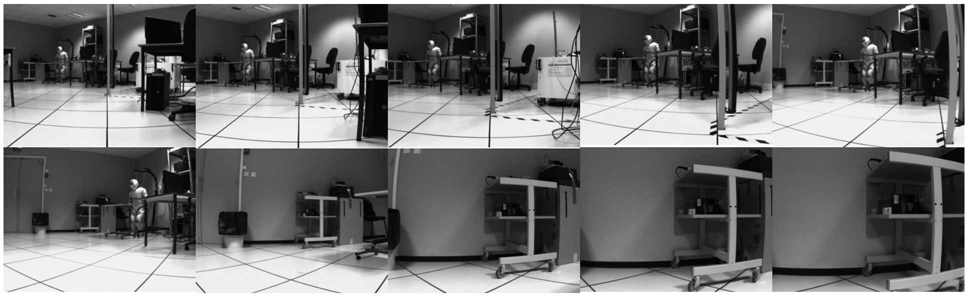


Figure 5.2 – Key images for navigation inside the robotics room.

5.3.2 Navigation

The robot was placed inside the mapped environment with the camera facing towards the mapped direction (Initial position shown by green dot in Figs. 5.4-5.7). The forward velocity was set to 0.08m/s and reduced to 0.05m/s when turning, whereas the rotational velocity was controlled by the navigation algorithm. During navigation, the robot has been able to follow the learned trajectory as shown by the blue curve in Figs. 5.4-5.7, with automatic switching of the key images.

5.3.2.1 Navigation inside a robotics room

Here we performed two experiments: the first one to validate our approach and the second one to show the robustness of our method in case of changes in illumination. In both

[1] http://www.irisa.fr/lagadic/team/Suman.Bista.html#videos.

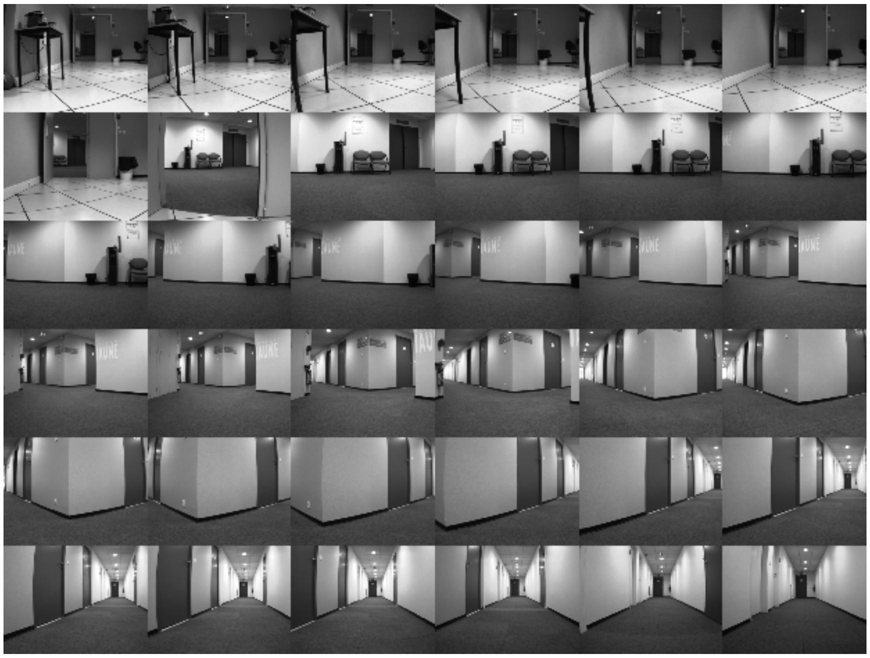


Figure 5.3 – Key images of the room and corridor.

cases, navigation has been performed with some changes in environment, like the table that is shown in the last key image in Fig. 5.2, was moved during the navigation. Fig. 5.4 shows the navigation of the Pioneer in the map without any change in illumination. The navigation in presence of change of illumination is depicted in Fig. 5.5, where all lights have been made dimmer. In both cases, our framework has been able to select the initial key frames and follow the learned path..

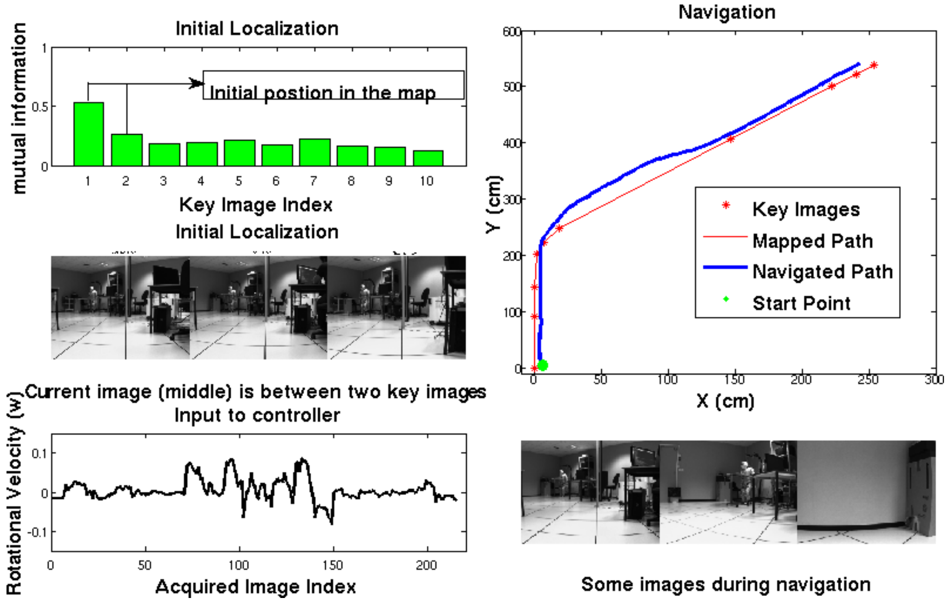


Figure 5.4 – Initial localization and navigation inside the robotics room.

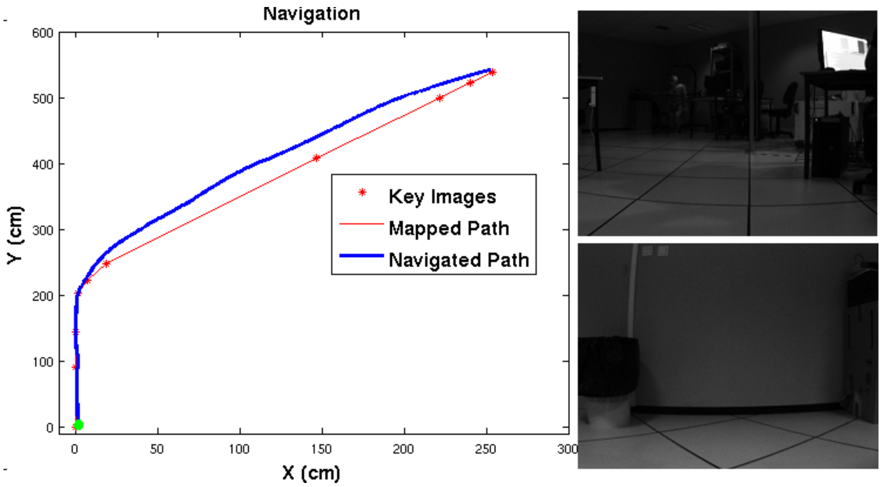


Figure 5.5 – Navigation inside the robotics room with change in illumination.

5.3.2.2 Navigation in a room and a corridor

Fig. 5.6 shows the navigation in the room and a corridor where the path consists of multiple turns. The robot moved from inside the room to the corridor. The robot followed the learned path with turning whenever it was required despite presence of moving people during navigation. Right angle turning is a challenging task especially in the corridors with similar/low texture. However, the key images obtained from the mapping part were still able to handle such situations. The lateral drift was within 7 cm from the mapped position. Indeed, our objective was just to perform a successful navigation without any accurate pose correction with the learned path.

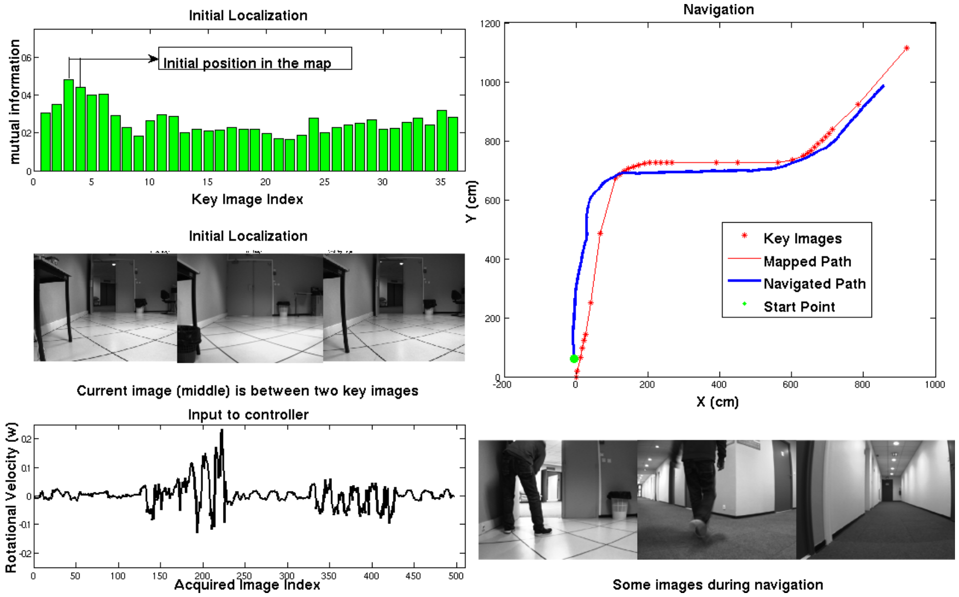


Figure 5.6 – Navigation between the room and the corridor.

5.3.2.3 Navigation in a corridor with peoples passing by

In this case, we performed the navigation in two corridors of length 30m and 31m. They are represented by 136 and 96 key images respectively as shown in Figs. 5.8 and 5.9, which were selected from the learning sequence of 6046 and 7507 images respectively. The second corridor consists of large windows that allow seeing outdoor so that robot undergoes change in illumination as it enters inside. Even in presence of moving people and change in illumination (Fig. 5.7), the robot was able to follow the learned trajectory.

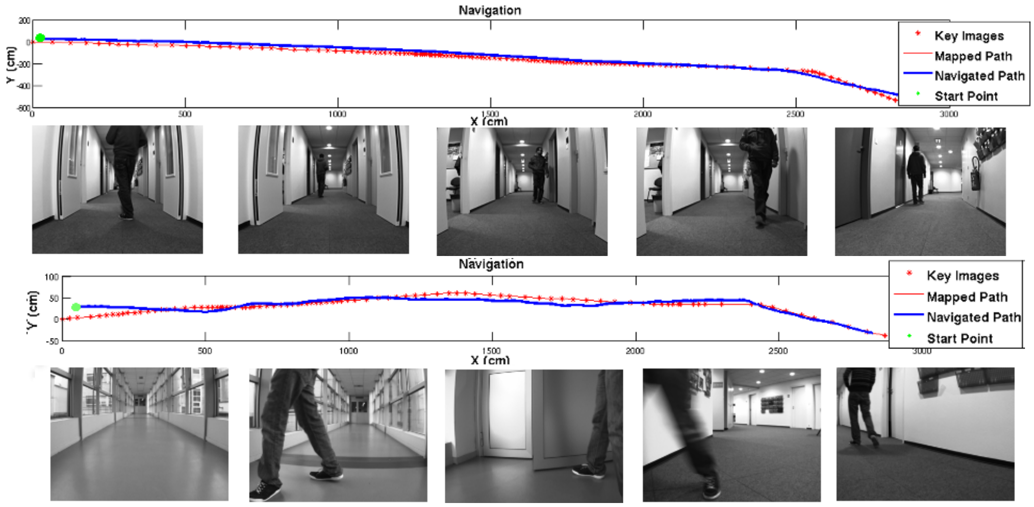


Figure 5.7 – Navigation in the corridors.

5.3.3 Discussion

The presented results show the viability of our approach in different scenarios and constraints. The robot has been able to follow autonomously the learned path from the start position. Our navigation framework is based upon the maximization of the mutual information, where the robot performs the navigation task to maximize the mutual information with the nearest key images with one degree of freedom. Based on this, the key images are switched automatically and the appropriate rotational velocity is set for allowing the robot to follow the learned path. IBVS has been able to keep the error within small bounds. However, there are still lateral deviations from the learned path. The possible reasons are: first, we do not perform any pose correction; second the path is approximated by straight lines; and last, the maximization of mutual information does not always guarantees global optimization. Still, successful navigation has been performed, which is our primary objective. Our initial localization is based upon the fact that the nearby key images share more entropy than the farther ones. The bar graphs in the Figs. 5.4 and 5.6 confirm this idea. Initial localization is key for successful navigation. Therefore, the images are registered at finer level before calculating mutual information. This process is time consuming. However, use

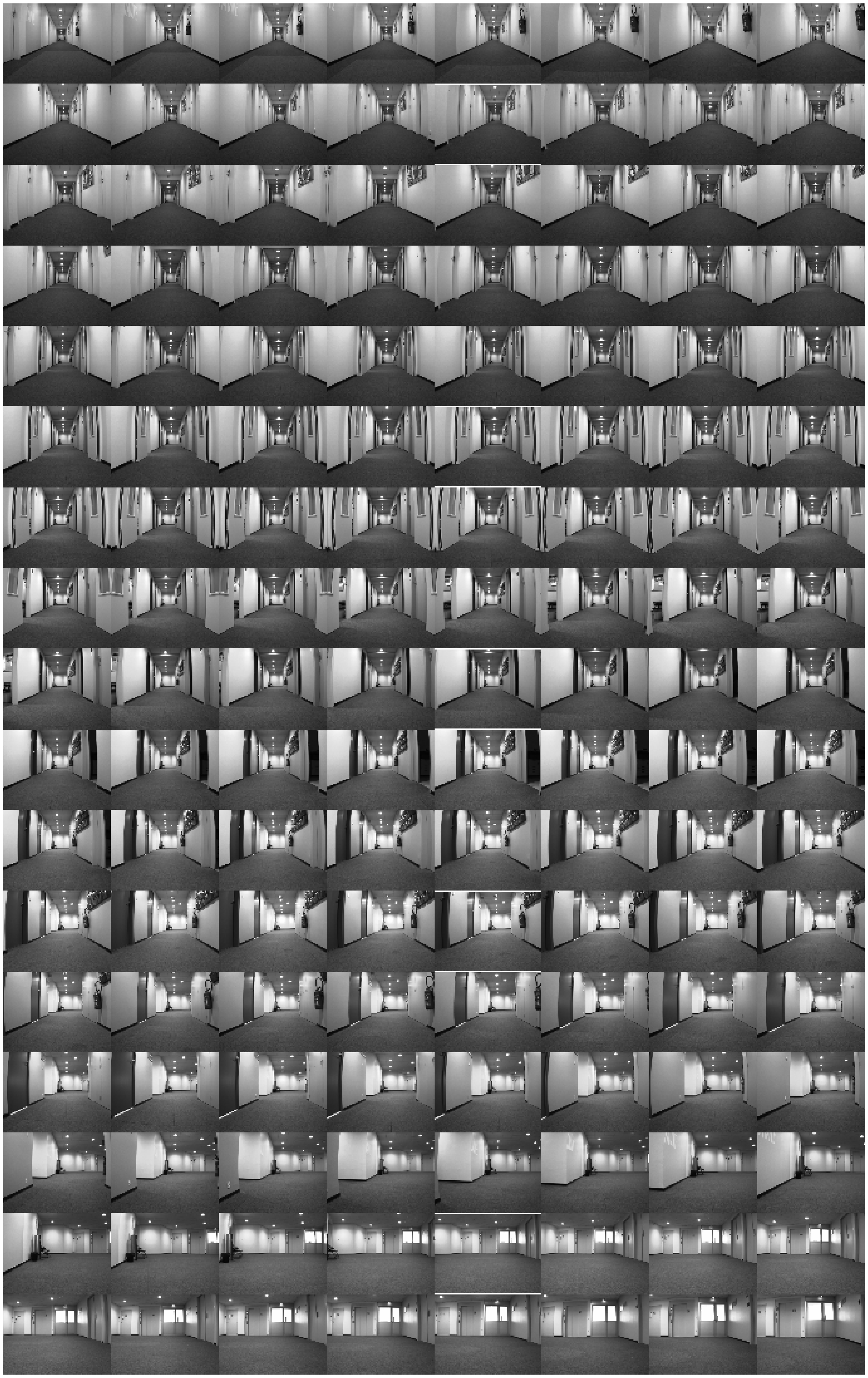


Figure 5.8 – Key images representing indoor corridor.

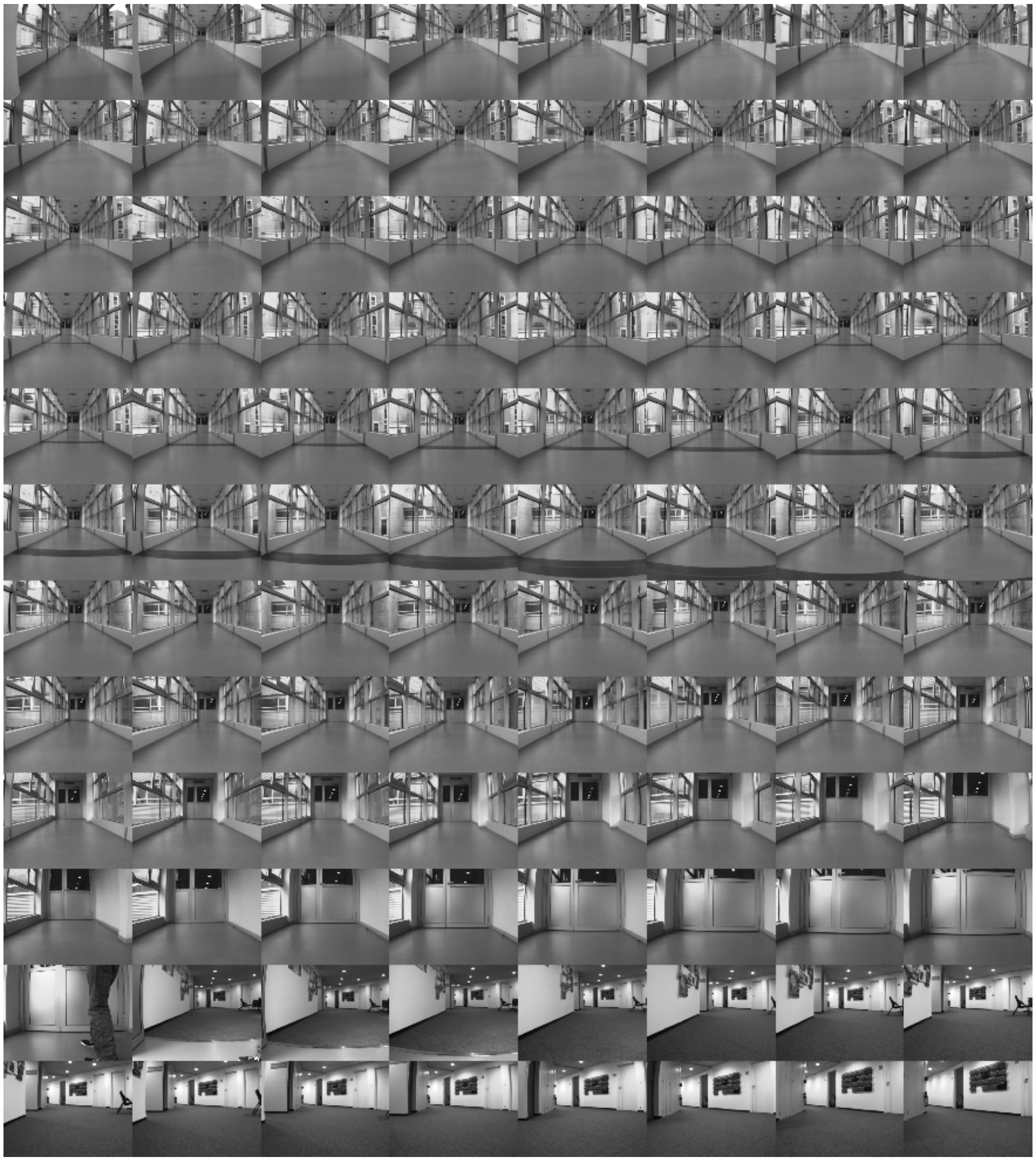


Figure 5.9 – Key images representing corridor with outdoor views.

of multiple pyramids (3 have been used in our experiments) and the use of translation transformation allow speeding up the process. After initial localization, the successive navigation has been done in real time at 5Hz.

Comparing with feature based methods, our method performs better where reliable point/-lines based features cannot be detected, resulting in failure in tracking/matching because the mutual information uses the entire image information and does not require feature detection. Besides that, mutual information is also able to handle changes in lightning conditions. However, our framework also has some limitations that are mainly due to poor conditioning of the mutual information especially in cases with few or no texture at all. Nevertheless, most of these problems can be greatly avoided by selecting a proper trajectory during the mapping. The other problem is that it is difficult to discriminate the two images when there is large occlusion and texture is too low or almost same.

5.4 Conclusions

We have presented a complete method for indoor qualitative mapping and navigation based on image memory using mutual information. Our navigation is exclusively based on image information (entropy) without relying on any 3D reconstruction, feature extraction, matching or tracking process. Using the information from the entire image makes navigation possible despite some level of occlusions (like people moving), blurs in the image, and changes in lighting conditions. Difficult situations include featureless areas like smooth/texture-less walls and rapid turnings. Besides that, it is very hard to discriminate between two places from mutual information which have similar histograms. Using whole image has advantage of not requiring feature extraction. But the feature-based methods are insensitive to changes outside of the tracked pattern, which we show in coming chapters. Using the entire image as input appears to reduce the robustness of the resulting system compared to feature/based methods. The rotational velocity was fed to the robot without any post processing and filtering, which sometimes resulted in non-smooth motion. More precise robot control strategies like filtering the velocity can address these issues.

Chapter 6

Appearance-based Indoor Navigation using Line Segments

This chapter presents a method for image-based navigation from an image memory using line segments as landmarks. The entire navigation process is based on 2D line segments without using any 3D information at all. The reference images are selected automatically in a prior learning phase based on line segments matching. The switching of reference images during the navigation is done exploiting also line segment matching between the current acquired image and nearby reference images. Three view matching result is used to compute the rotational velocity of a mobile robot during its navigation by IBVS. As previously, real-time navigation has been validated inside a corridor and inside a room with a Pioneer 3DX equipped with an on-board camera. The obtained results confirm the viability of our approach, and verify that accurate mapping and localization are not necessary for a useful indoor navigation as well as that line segments are better features in structured indoor environment w.r.t. feature points [ŠRDC09] and Mutual Information (Chapter 5).

The work described in this chapter has been published in [BRGC16a] and presented at ICRA-2016.

6.1 Line Segments as Feature for Indoor Navigation

Mutual information-based navigation only works better in environments that have good textures. So, they have poor discriminating power and high perceptual aliasing effect. Like other global features, MI-based navigation does not cope with large occlusions. However, with local features like points ([ŠRDC09, DSRC11]), these problems are mitigated. Robots are able to navigate in urban environments and in all places where local point based features are abundant. Because of windows, wiry structures, reflections and repetitions, as well as limited texture in indoor scenarios, most of the techniques employed in outdoor environments typically result in a significant performance drop when applied to indoor scenarios, which causes standard procedures based on image descriptors to poorly perform indoors [MW15]. A typical navigation task in an indoor environment can be divided into two parts:

a) Navigation through corridors and

b) Navigation inside rooms.

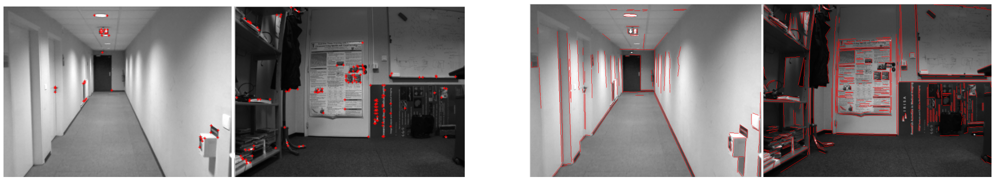


Figure 6.1 – Points and line segments detection in indoor environment

For the latter case, it is more likely to have abundant distinctive local features and global features whereas, for the former case, the perceived surface may not give enough features points for navigation. Moreover, similar texture or lack of texture may result in false matching. However, in indoor environments, line segments are abundant. In addition to this, line segments are more robust to partial occlusions and more resilient to motion blur [ZK11, DYO05]. Also, line segments in the image can be detected accurately in real time by line algorithms like LSD [GJMR10] and ED Lines[AT11]. Because of all these reasons, this chapter explores the use of line segments as visual landmarks.

Tracking/matching of multiple line segments is still an open problem in computer vision. This is due to inaccurate locations of line endpoints, fragmentation of lines, lack of strongly disambiguating geometric constraints for an image pair, and lack of distinctive appearance in low-texture scenes [SZ00, FWH10, ZK13]. Despite these problems, [GMC06, ZK11, SRD06] have demonstrated line segments-based navigation, but using a 3D model-based approach. In [GMC06], a model-based SLAM using 3D lines as landmarks has been presented, where unscented Kalman Filters are used to initialize new line segments and generate a 3D wire-frame model of the scene that can be tracked with a robust model-based tracking algorithm. The authors of [SRD06] have extended the monocular SLAM using points [DRMS07] to line segments, where Kalman Filters are used to track the lines. Both methods rely on control points (a set of sample points placed along the line) for tracking, which, however, is not suitable when line segments are close to each other because of failure in tracking. Recently [ZK11] has used Nearby Line Tracking to track lines and an EKF is used to predict and update the state of the camera and line landmarks.

In [FOPV13, PKB14, VSSV00], a vision-based corridor navigation algorithm has been proposed that uses the vanishing point extracted from corridor guidelines for the Nao humanoid robot, a wheelchair and a mobile robot respectively. The first two works are mapless methods. In [VSSV00], the vanishing point is used for the heading control whereas an appearance-based process is used to monitor the robot position along the path. A set of reference images are acquired manually at relevant positions along the path which correspond either to areas in the workspace where some special action can be undertaken (e.g. doors, elevators, corners, etc.) or viewpoints where very distinctive images can be acquired. During navigation, these reference images are compared with current images using the SSD metric. The solution in [DYO05] not only uses two pairs of natural line and point, but also the odometer data (to determine the height of the landmarks) for the visual localization.

Our Contribution

Our main contribution is a complete method for indoor navigation (automatic construction of a navigation route, initial localization that enables the robot to start from any position within the map, successive localization and a control law for choosing the rotational velocity) that coarsely follows the learned path by just using the information provided by the 2D line segments detected in the image *without* need of accurate mapping, localization and robot odometry. Our method does not depend upon any specific types of lines (e.g. vertical lines or corridor lines). Indeed, we show that the information obtained from the 2D line segment matching between the current acquired image and nearby reference images is enough for automatic switching of key images and for robot control *without* 3D reconstruction.

6.2 Our Navigation Framework

6.2.1 Line Segments Detection and Matching

In our work, EDLines detector [AT11] has been used to detect lines and the algorithm proposed by [ZK13] has been used to generate pairwise line correspondences. These methods have been selected because of their high accuracy and computational speed.

For two views, the line segments do not provide strong geometric constraints as opposite to points (epipolar geometry). The trifocal tensor provides instead a strong geometric constraint for lines, but it requires line correspondences in three views (Ref. Sect. 2.1.3.3). The estimation of the trifocal tensor with RANSAC [HZ03] can be used to verify the line segment correspondences in three views. However, the cost function associated with the trifocal tensor is computationally more expensive than in the fundamental matrix case when used with RANSAC. In addition, at least 13 line correspondences are required to compute the trifocal tensor (instead of only 6 point correspondences) for three views. Nevertheless, the number of outliers in three views matching is quite low compared to two views only, which makes it possible for the RANSAC-based estimation to converge in few iterations. The process is discussed in Sect. 2.1.3.4.

In our method, two view matches are used in initial localization, switching of key images and generating three view correspondences. Three view matches are used in mapping, switching of key images and motion control. For three view matching, the current key image and the two most recently acquired images are used during the mapping, whereas, the two key images and the currently acquired image are used during the navigation. When obtaining the three view correspondences, only the matched lines between the first two images are used to match with the third image in order to reduce the cost of matching (see Fig. 6.2).

6.2.2 Mapping from Line Segments

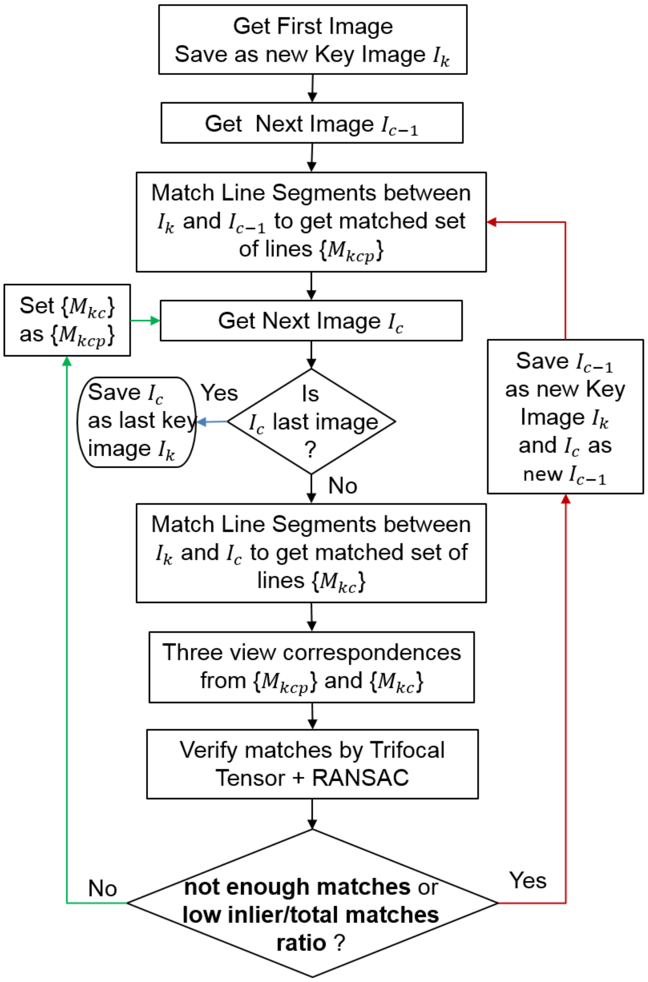

Get First Image
Save as new Key Image I_k
Get Next Image I_{c-1}
Match Line Segments between I_k and I_{c-1} to get matched set of lines $\{M_{kcp}\}$
Set $\{M_{kc}\}$ as $\{M_{kcp}\}$
Get Next Image I_c
Save I_{c-1} as new Key Image I_k and I_c as new I_{c-1}
Save I_c as last key image I_k
Yes
Is I_c last image ?
No
Match Line Segments between I_k and I_c to get matched set of lines $\{M_{kc}\}$
Three view correspondences from $\{M_{kcp}\}$ and $\{M_{kc}\}$
Verify matches by Trifocal Tensor + RANSAC
No
not enough matches or low inlier/total matches ratio ?
Yes

Figure 6.2 – Building the map from line segments.

The key image selection procedure is sketched in Fig. 6.2. Let I_{c-1} and I_c be the two most recently acquired images and I_k be the most recent key image. For the case just after the new key image is set, I_{c-1} and I_c are the two images acquired successively after I_k. The detected line segments of I_k are matched with I_{c-1} to get the first set of matched lines $\{M_{kcp}\}$. The lines in I_k present in $\{M_{kcp}\}$ are matched with the detected line segments in I_c to get the second set of matched lines $\{M_{kc}\}$. The common line segments in $\{M_{kcp}\}$

and $\{M_{kc}\}$ give three view correspondences. If there are not sufficient number of lines (for example less than 20) after three view matching, or a low ratio (for example less than 0.5) of inlier to total number of matches after trifocal tensor estimation with RANSAC, I_{c-1} is saved in the database as a recent key image I_k, and I_c becomes the new I_{c-1}. Otherwise, $\{M_{kc}\}$ becomes $\{M_{kcp}\}$ and the line segments of next acquired image I_c and I_k are matched to get a new set of $\{M_{kc}\}$. This idea is similar for tracking the line segments of the key image in successive frames. Three view matching is always done between the current key image I_k and the two most recently acquired images I_{c-1} and I_c. Hence, the output of the mapping process is a set of key images that represents the arc the robot has to follow during the navigation. The neighboring key images share some common line segments as shown in Fig. 6.3, which makes it possible to consider multiple key images in a neighborhood for defining the heading angle of the robot.

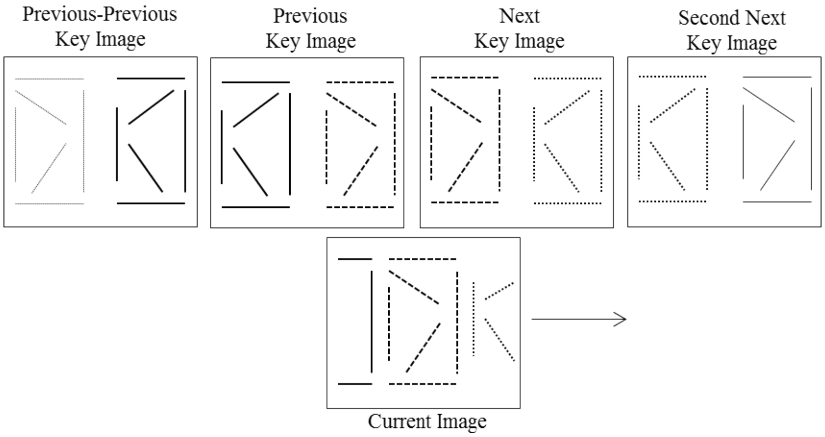


Figure 6.3 – The map consists of key images and line segments. Adjacent key images share some line segments with the current image. These corresponding line segments with the current acquired image are used for motion control with the aim of following the arc defined in the map.

6.2.3 Navigation in the Map

6.2.3.1 Qualitative Localization

Initial Localization The navigation starts with the initial localization where the first image acquired (I_a) is compared with all the images in the database based upon line segment matching. The key image with the maximum number of matches is selected. Let it be I_{k_i}. Then the adjacent key image with second maximum matches is also selected. This image can be either $I_{k_{i+1}}$ or $I_{k_{i-1}}$. If the robot is assumed to be moving along the same direction as the images arranged in the database, I_a is between $I_{k_{i-1}}$ and I_k, or I_k and $I_{k_{i+1}}$. For simplicity, we denote the previous key image as I_P and the next key image as I_N. If $n(...)$ is the number of lines matched between the images, the initial localization process can be expressed as:

$$I_{k_i} = \arg\max_{I_k}\{n(I_a,I_{k_1}),n(I_a,I_{k_2}),......,n(I_a,I_{k_n})\}$$

$$I_{k_j} = \arg\max_{I_k}\{n(I_a,I_{k_{i-1}}),n(I_a,I_{k_{i+1}})\}$$

$$I_P = I_{k_j} \text{ and } \quad I_N = I_{k_i}\text{if } i > j$$
$$I_P = I_{k_i} \text{ and } \quad I_N = I_{k_j}\text{if } i < j$$

Successive Localization After initial localization in the map, further localizations can be done by just comparing with few adjacent images in the database. The previous key image I_P, the next key image I_N and the second next key image I_{NN} are compared with the current acquired image I_a. Let $n(...)$ be the number of lines matched between the images. Then switching of key images is done when at least one of the following criteria is fulfilled for two consecutive acquired images I_a and I_{a+1}:

$$n(I_a,I_N,I_{NN}) > n(I_P,I_a,I_N) \text{ or}$$

$$n(I_a,I_{NN}) > n(I_a,I_N) \ \&\& \ n(I_a,I_{NN}) > n(I_p,I_a).$$

The first criterion is based on the result of three view matching between the images inside the brackets, whereas the second criterion is based on two view matching of images. The second criterion is essentially useful when there are no three view matches or very few number of three view correspondences. Such a condition may sometimes occur with sharp turns in corridors having no texture at all. After switching the images, I_N becomes I_P, I_{NN} becomes I_N, and next key image from I_N becomes I_{NN}. Then the process repeats. When the end of the database is reached, I_{NN} will not be available and I_N will be the last image acquired during the mapping. So, the navigation needs to be stopped. Otherwise, the robot will be moving out of the mapped environment.

6.2.3.2 Motion Control

For navigation, we control rotational velocity to define the heading angle of the robot. Basically, we follow the same approach as explained in Sect. 4. The rotational velocity is derived from the matched lines between I_a, I_N and I_{NN}, whereas the translational velocity is kept constant and reduced to smaller constant value when turning. Such turnings are automatically detected by looking at the commanded rotational velocity.

In order to drive current featuress to desired ones $\mathbf{s}^*$, we control ω_r as [CH06]

$$\omega_r = -\mathbf{J}_\omega^+(\lambda(\mathbf{s}-\mathbf{s}^*)+\mathbf{J}_v v_r), \qquad (6.1)$$

where λ is a positive gain, $\mathbf{J}_v$ and $\mathbf{J}_\omega$ are the Jacobian associated with v_r and ω_r respectively and $\mathbf{J}_\omega^+$ is the pseudo-inverse of $\mathbf{J}_\omega$. The expression of $\mathbf{J}_v$ and $\mathbf{J}_\omega$ can be obtained as follows.

In 3D, a straight line can be represented by the intersection of two planes:

$$a_i x + b_i y + c_i z + d_i = 0,\ i = 1,2. \tag{6.2}$$

Except for the degenerate cases ($d_1 = d_2 = 0$), a 3D line in a scene projects onto the image plane as a 2D line. As in [ECR92], we choose to parameterize line segments with parameters (ρ, θ) as

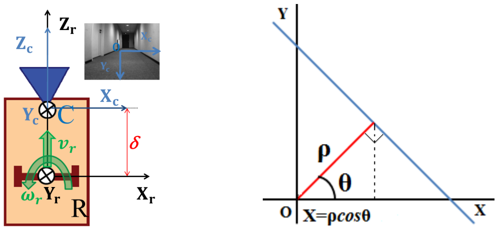


Figure 6.4 – Top view of robot (orange) equipped with a perspective camera (blue) with its optical axis perpendicular to axis of robot rotation, (Left) and Representation of line in polar form (Right).

$$X\cos\theta + Y\sin\theta - \rho = 0. \tag{6.3}$$

The interaction matrix related to ρ and θ is given by [ECR92]

$$\begin{array}{ll} L_\rho = & [\lambda_\rho\cos\theta \quad \lambda_\rho\sin\theta \quad -\lambda_\rho\rho \quad (1+\rho^2)\sin\theta \quad -(1+\rho^2)\cos\theta \quad 0] \\ L_\theta = & [\lambda_\theta\cos\theta \quad \lambda_\theta\sin\theta \quad -\lambda_\theta\rho \quad -\rho\cos\theta \quad -\rho\sin\theta \quad -1], \end{array} \tag{6.4}$$

where $\lambda_\rho = (a_i\rho\cos\theta + b_i\rho\sin\theta + c_i)/d_i$ and
$\lambda_\theta = (a_i\sin\theta - b_i\cos\theta)/d_i$.

From (4.6) and (6.4), we can obtain following expression w.r.t robot velocity (v_r, ω_r) as

$$\begin{bmatrix} \dot\rho \\ \dot\theta \end{bmatrix} = \begin{bmatrix} -\lambda_\rho\rho & -(1+\rho^2)\cos\theta - \delta\lambda_\rho\cos\theta \\ -\lambda_\theta\rho & -\rho\sin\theta - \delta\lambda_\theta\cos\theta \end{bmatrix} \begin{bmatrix} v_r \\ \omega_r \end{bmatrix}. \tag{6.5}$$

Since we only control ω_r, only one feature derived from all line segments is sufficient. We have chosen the abscissa of the centroid of the points of intersection of the matched lines and their respective normal from the origin. For a given line as shown in Fig. 6.4 (right), $X = \rho\cos\theta$ gives the abscissa of such a point.

For a set of n matched lines between I_a, I_N and I_{NN}, we define:

$$\begin{array}{rr} X_{ai} = \rho_{ai}\cos\theta_{ai}, & X_a = \frac{1}{n}\sum_{i=1}^{n} X_{ai}, \\ X_{Ni} = \rho_{Ni}\cos\theta_{Ni}, & X_N = \frac{1}{n}\sum_{i=1}^{n} X_{Ni}, \\ X_{NNi} = \rho_{NNi}\cos\theta_{NNi}, \ \text{and} & X_{NN} = \frac{1}{n}\sum_{i=1}^{n} X_{NNi}. \end{array} \tag{6.6}$$

Hence, our visual feature is $s = X_a$ and the desired feature is $s^* = X_N$. We then have

$$s = X_a = \frac{1}{n}\sum_{i=1}^{n} \rho_{ai} \cos\theta_{ai}.$$

Taking derivative w.r.t. time , we obtain

$$\dot{s} = \dot{X}_a = \frac{1}{n}\sum_{i=1}^{n} (\dot{\rho}_{ai}\cos\theta_{ai} - \rho_{ai}\sin\theta_{ai}\dot{\theta}_{ai}). \tag{6.7}$$

From (6.5), we can deduce (6.7) as

$$\begin{aligned}\dot{s} = \; & \tfrac{1}{n}\sum_{i=1}^{n}((\lambda_{\theta ai}\rho_{ai}^2 \sin\theta_{ai} - \lambda_{\rho ai}\rho_{ai}\cos\theta_{ai})v_r + ((1+\rho_{ai}^2)\cos^2\theta_{ai} - \rho_{ai}^2\sin^2\theta_{ai} \\ & + \delta(\lambda_{\theta ai}\rho_{ai}\sin\theta_{ai}\cos\theta_{ai} - \lambda_{\rho ai}\cos^2\theta_{ai}))\omega_r).\end{aligned} \tag{6.8}$$

On simplification,

$$\begin{aligned}\dot{s} = \; & \frac{1}{n}\sum_{i=1}^{n}(\rho_{ai}(\lambda_{\theta ai}\rho_{ai}\sin\theta_{ai} - \lambda_{\rho ai}\cos\theta_{ai})v_r + \\ & \delta\cos\theta_{ai}(\lambda_{\theta ai}\rho_{ai}\sin\theta_{ai} - \lambda_{\rho ai}\cos\theta_{ai}))\omega_r).\end{aligned} \tag{6.9}$$

From (4.7) and (6.9), we obtain

$$\begin{aligned}J_v &= \tfrac{1}{n}\sum_{i=1}^{n}(\lambda_{\theta ai}\rho_{ai}^2\sin\theta_{ai} - \lambda_{\rho ai}\rho_{ai}\cos\theta_{ai}). \\ J_\omega &= \tfrac{1}{n}\sum_{i=1}^{n}(\cos^2\theta_{ai} + \rho_{ai}^2\cos 2\theta_{ai} + \delta\cos\theta_{ai}(\lambda_{\theta ai}\rho_{ai}\sin\theta_{ai} - \lambda_{\rho ai}\cos\theta_{ai})).\end{aligned} \tag{6.10}$$

Neglecting δ with respect to d_i (distance of line from image plane), and assuming the camera optical axis is orthogonal to the axis of robot rotation and that the centroid stays near the image plane center , (6.10) can be approximated as

$$J_v \simeq 0 \text{ and } J_\omega \simeq \frac{1}{n}\sum_{i=1}^{n}(\cos^2\theta_{ai} - \rho_{ai}^2\cos(2\theta_{ai})) = J_a,\ \delta \ll d_i, \rho_{ai} \ll d_i, \rho_{ai}^2 \ll d_i. \tag{6.11}$$

Since visual servoing is known to be robust against modeling errors [CH07], such approximations are reasonable. Thus, from (6.1) and (6.11) we finally obtain the following expression for the rotational velocity:

$$\omega_r = -\frac{\lambda}{J_a \pm \varepsilon}(X_a - X_N), \tag{6.12}$$

where ε is a small constant to prevent possible division by zero. In order to smooth the rapid steering actions when switching between frames, a feed-forward command is also added to ω_r. The calculation of the feed-forward term is based on the difference of the centroids between the shared lines of I_a with I_N and I_{NN}. The final equation is given as follows

$$\omega_r = -\frac{\lambda}{J_a \pm \varepsilon}(h_1(X_a - X_N) + h_2(X_a - X_{NN})), \tag{6.13}$$

where h_1 and h_2 are positive weights such that $h_1 + h_2 = 1$.

Thus, our complete framework uses only the *2D* information obtained from *line segments* matching, without requiring any 3D information, which was not the case in previous works. From this 2D information, we derive the required rotational velocity using *IBVS*, which makes the robot to follow the learned path successfully without the need of any accurate mapping or localization.

Stability Analysis

If J_ω is the true Jacobian and $\hat{J}_\omega$ is the approximated Jacobian related to ω_r, from (6.10) and (6.11) we have

$$J_\omega = \frac{1}{n}\sum_{i=1}^{n}(\cos^2\theta_{ai} + \rho_{ai}^2\cos 2\theta_{ai} + \delta\cos\theta_{ai}(\lambda_{\theta ai}\rho_{ai}\sin\theta_{ai} - \lambda_{\rho ai}\cos\theta_{ai})). \tag{6.14}$$

$$\hat{J}_\omega \simeq \frac{1}{n}\sum_{i=1}^{n}(\cos^2\theta_{ai} - \rho_{ai}^2\cos(2\theta_{ai})). \tag{6.15}$$

Therefore,

$$\frac{J_\omega}{\hat{J}_\omega} = 1 + \frac{\frac{1}{n}\sum_{i=1}^{n}\delta\cos\theta_{ai}(\lambda_{\theta ai}\rho_{ai}\sin\theta_{ai} - \lambda_{\rho ai}\cos\theta_{ai})}{\frac{1}{n}\sum_{i=1}^{n}(\cos^2\theta_{ai} - \rho_{ai}^2\cos(2\theta_{ai}))} > 0, \tag{6.16}$$

since $\delta \ll d_i$ and λ_{xai} is function of $(1/d_i)$. From (6.11), we have Jacobian related to v_r is close to zero. Hence, with (6.16), the global asymptotic stability is guaranteed as inferred by (4.19).

6.3 Results and Discussions

The mapping was done offline, whereas the navigation experiment was performed online at 6 Hz. For line segments detection and matching, the implementation provided by the MIP group [1], University of Kiel, has been used with some modifications as per our requirements. The qualitative results of mapping and navigation using different trajectories in corridor and inside the room are presented below. The videos of the experiments are available in [2]

[1] http://www.mip.informatik.uni-kiel.de/tiki-download_file.php?fileId=1965 [Accessed: August 24,2015].

[2] http://www.irisa.fr/lagadic/team/Suman.Bista.html#videos.

6.3.1 Experiment I: Inside a room

6.3.1.1 Mapping

617 images have been acquired as the learning sequence. 18 images shown in Fig. 6.5 have been selected automatically from the mapping algorithm described in Sect. 5.2.2 as key frames. The trajectory obtained from the odometry is shown by a red curve in Figs. 6.6 and 6.7, where the red symbol $*$ represents the location of the key images. The obtained key images are able to represent the learned path. There are more key images over a small distance in case of quick displacements of features like in turnings or when line segments cannot be successively matched over the sequence due to changes in illumination.

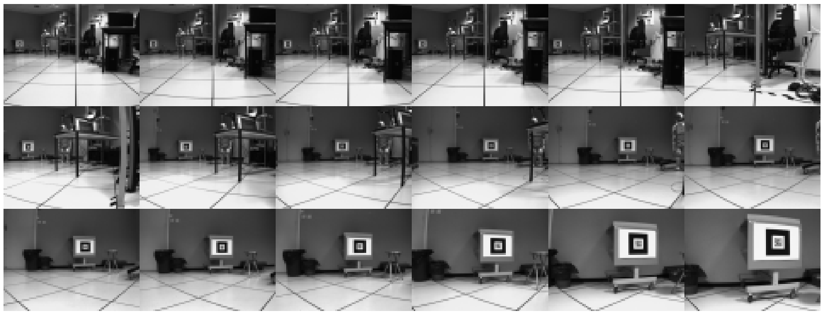


Figure 6.5 – Key images of the robotics room.

6.3.1.2 Navigation

The robot was placed inside the mapped environment with the camera facing towards the mapped direction (initial position shown by green dot). The forward velocity was set to 0.2 m/s. During navigation, the robot was able to follow the learned trajectory as shown by the blue curve in Figs. 6.6 and 6.7, with automatic switching of the reference images. Figure 6.6 shows the navigation of the Pioneer in the map without any change in environment from the time of mapping. The navigation in presence of obstacles is shown in Fig. 6.7 (left). Even during a continuously obstructed view by walking in front of the camera (as shown in Fig. 6.8 (left)), the robot was still able to follow the desired path. Fig. 6.7 (middle) shows the navigation with some changes in the room as shown in Fig. 6.8 (third column), where the table was moved from the end to the middle of the room and replaced by a chair and the stool was pushed further from the time of mapping.Fig. 6.7 (right) shows navigation in the room performed 6 months later than the mapping stage with many changes and dynamic objects in the scene like a table, chairs, boxes, etc (Fig. 6.8 (right)). The successful navigation in these latter cases was possible due to the presence of sufficiently large number of line matches from the static objects like ceilings, floor tiles, posters, and pillars. In all cases, the drift was within 3cm from mapped position.

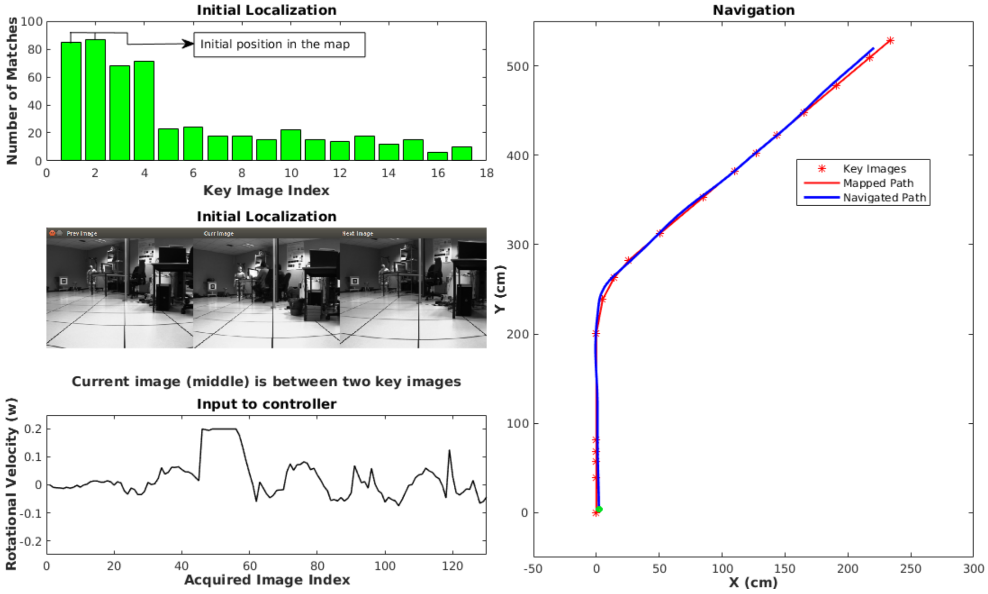


Figure 6.6 – Initial localization and navigation inside the robotics room.

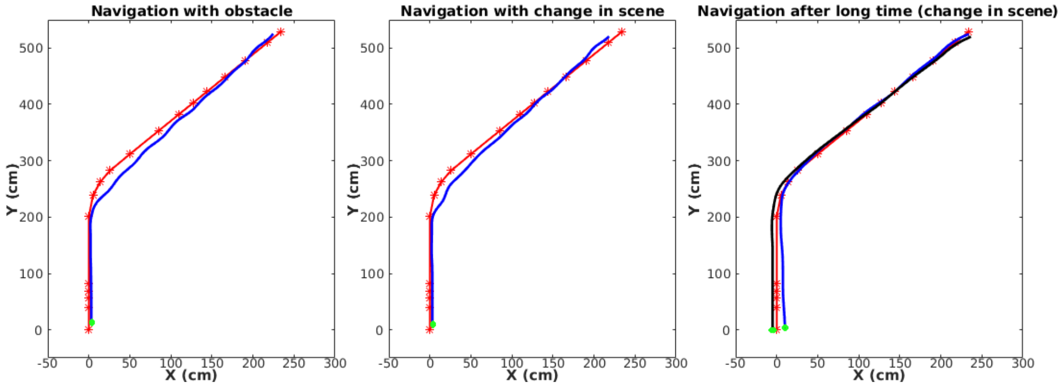


Figure 6.7 – Navigation inside the robotics room in presence of people (left) and changes in mapped scene (middle and right).

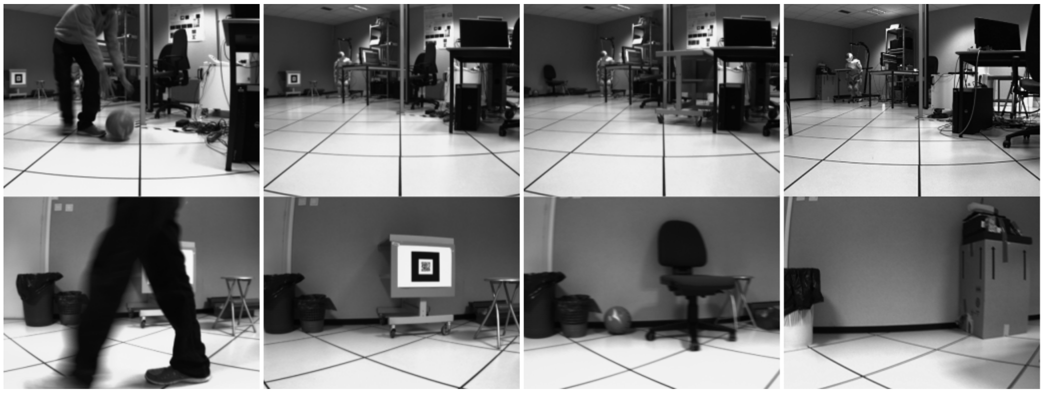


Figure 6.8 – Obstruction in view (left), mapped scene (second column) and changes in scene (third column, right).

6.3.2 Experiment 2: In a Corridor

6.3.2.1 Mapping

Out of 1208 images acquired in the corridor, 45 have been selected automatically as the key images (Fig. 6.9). Similarly, 53 key images have been obtained from 1083 images of the same corridor taken from a reverse direction (Fig. 6.10).

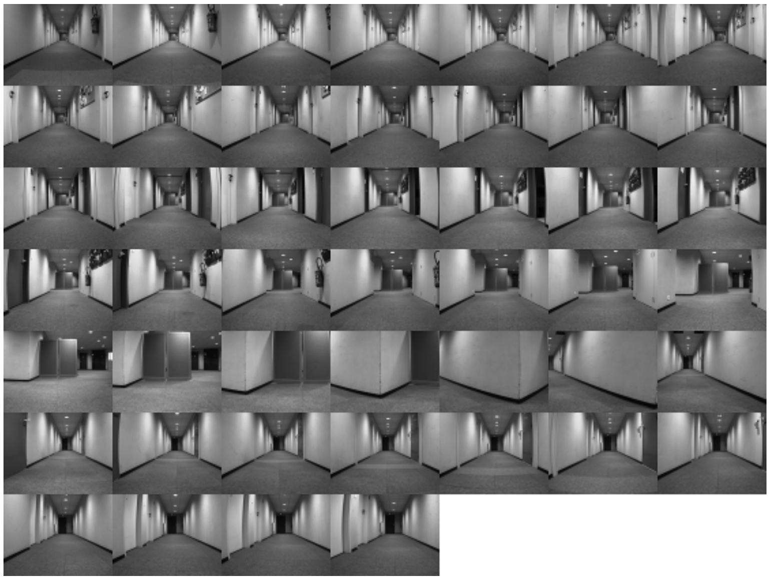


Figure 6.9 – Key images of the corridor.

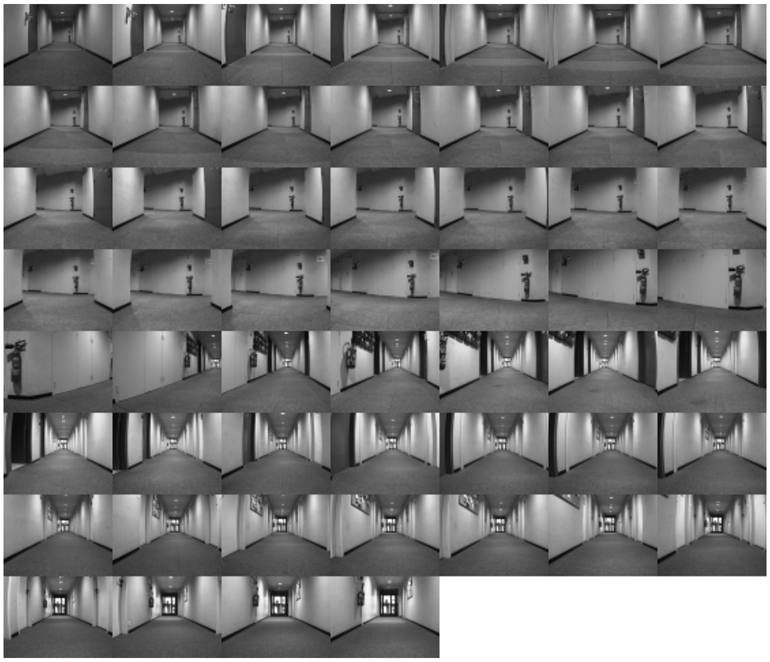


Figure 6.10 – Key images of the corridor from reverse direction.

The mapping has been done with all doors closed except one. This was meant to ensure that illumination from the room and outside windows has negligible effects. The obtained key images represent a path of length 32 meters. The distribution of key images concentrated at the turnings and when line segments of key images cannot be successively matched over the sequence.

6.3.2.2 Navigation

Figures 6.11 and 6.12 show navigation in the corridor. The robot was placed inside the mapped location (initial position shown by green dot). The forward velocity was set to 0.15m/s and reduced to 0.075m/s when turning, whereas the rotational velocity was controlled by the navigation algorithm. Even with the open doors, people walking in the corridor and blur in some images as shown in Fig. 6.13, the robot was still able to navigate successfully with turning whenever it was required. Right angle turning is a challenging task, in the sense that there are few lines with fast changes. However, the key images obtained from the mapping part were still able to handle such situations. The lateral drift when navigating through 32 meters in the corridor is within 5 cm from the mapped position, thus confirming the accuracy of the visual servoing control law.

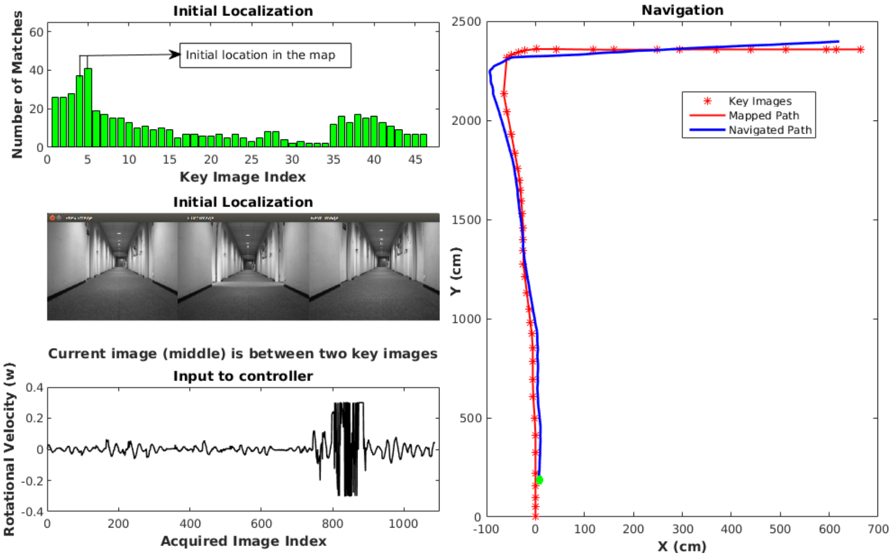


Figure 6.11 – Navigation in the corridor.

6.3.3 Experiment 3: Navigation in room and corridor

Two experiments have been performed in which the robot moves between a corridor and a room. In the first experiment, the robot navigated a 40m path from corridor to inside the room. Out of 7745 images acquired during mapping, 105 images have been automatically selected as reference images (Fig. 6.15). In the second experiment, the robot navigated

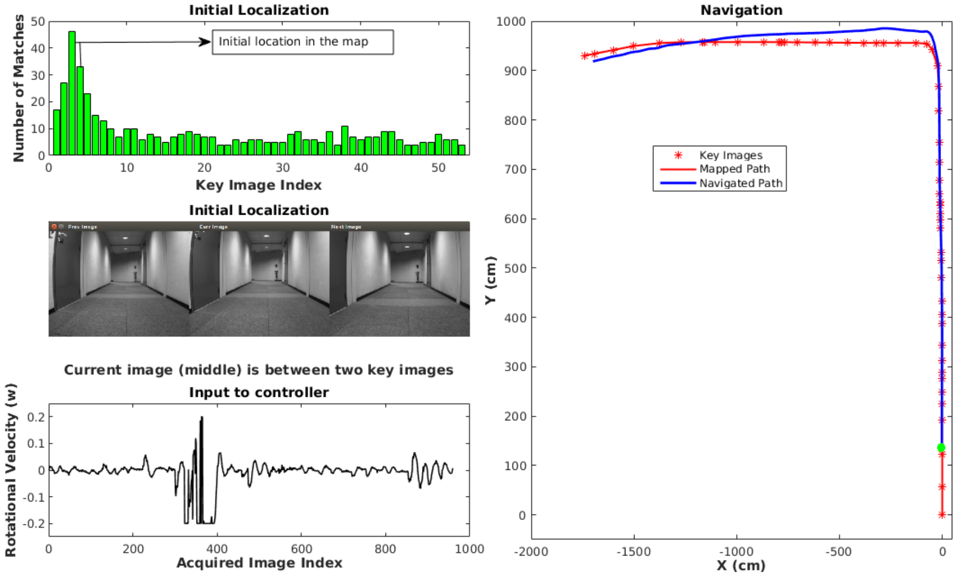


Figure 6.12 – Navigation in the corridor in reverse direction.

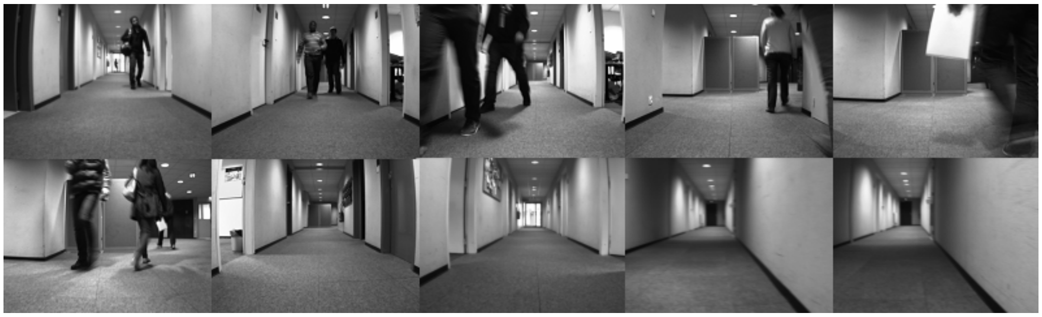


Figure 6.13 – Changes in the navigation environment from the mapped one: people passing by, opening of the doors, change in the local illumination due to a dead bulb and blur in the image.

inside the room and then into the corridor in a 22m path. Out of 4291 images acquired, 90 images were selected as reference images (Fig. 6.16). The navigation path consists of straight line and multiple turns as shown in Figs. 6.14-6.16.

Fig. 6.14 presents the navigation of the robot. The robot successfully followed the learned paths with turning whenever required. There are more deviations in turnings especially in case of turning in large angle (semi-circular turnings) because of approximation of the arcs as straight lines and few lines detected with fast changes between the frames. Even though a large drift is present while doing circular turn in the second experiment, the navigation was still successful.

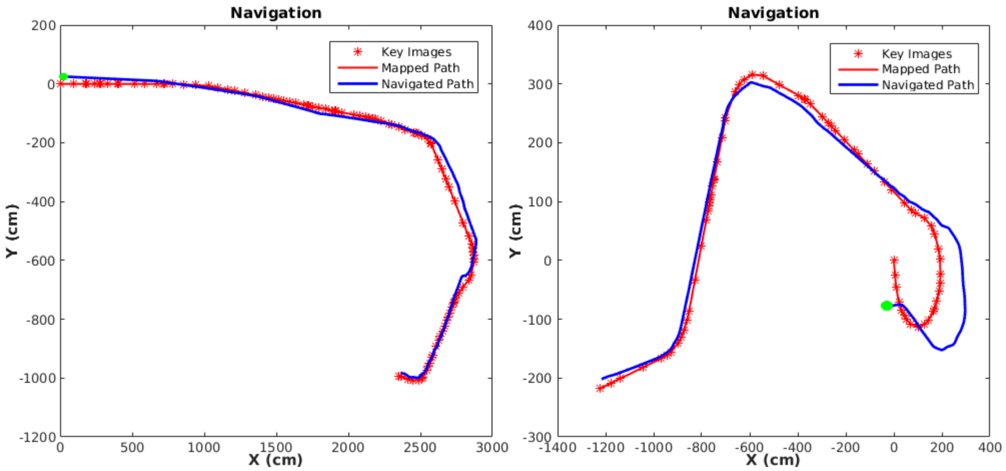


Figure 6.14 – Navigation in the corridor and the room.

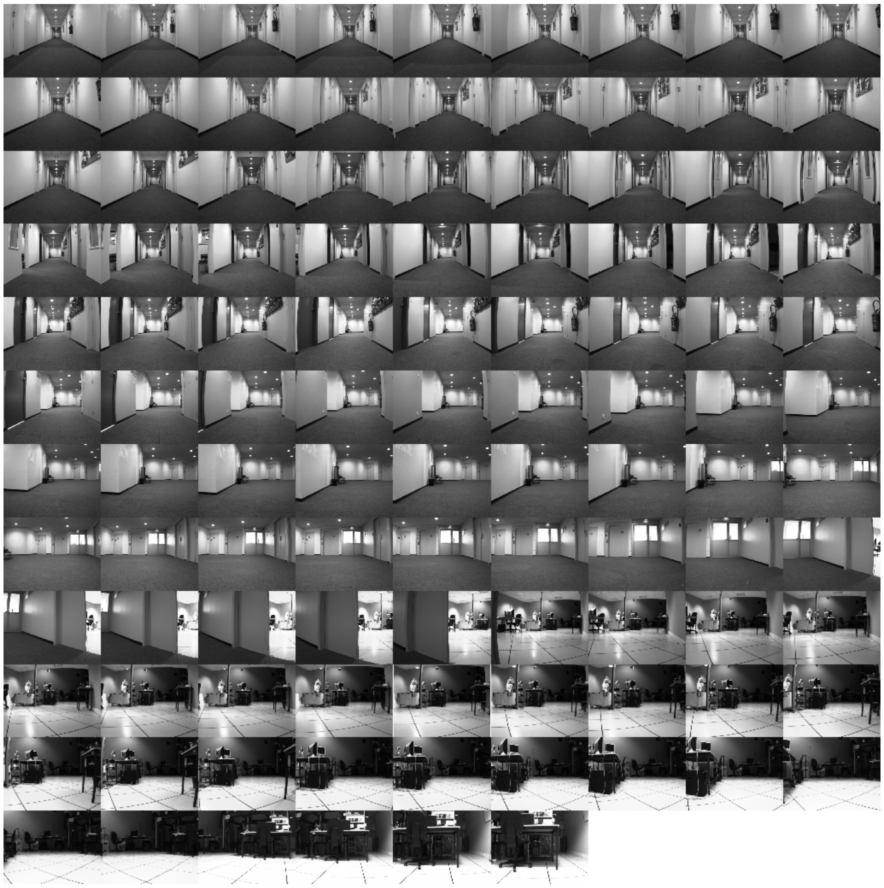


Figure 6.15 – Key images of 40m navigation path.

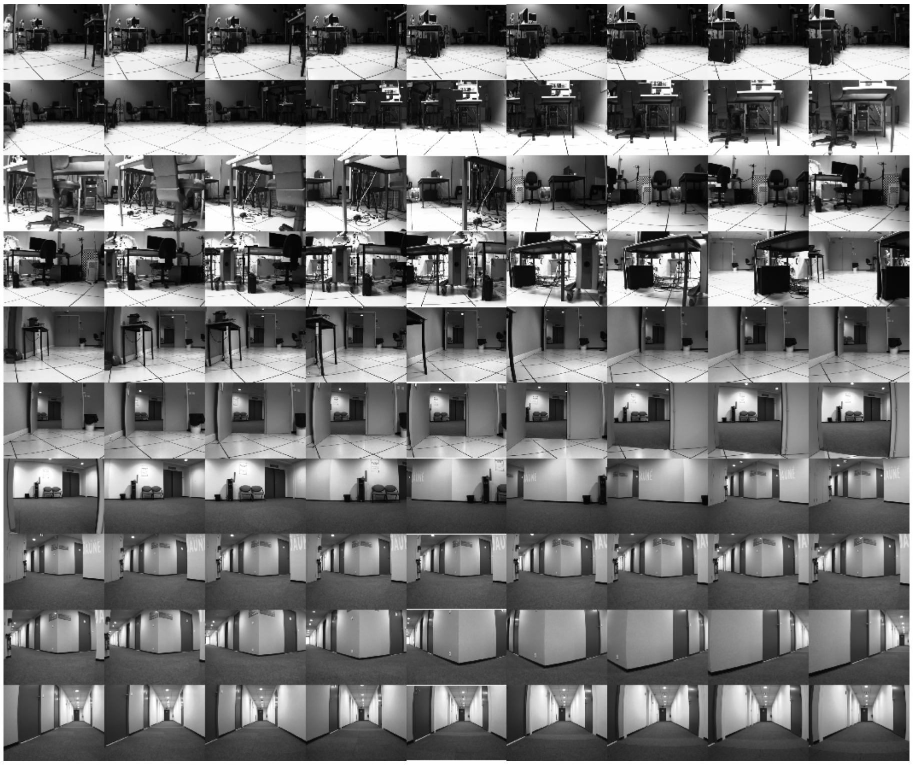


Figure 6.16 – Key images of 22m navigation path.

6.3.4 Discussions

The presented results show the viability of our approach in many different scenarios and constraints. The robot has been able to navigate autonomously in the learned path from the start position. Our framework does not depend upon any particular type of line segment, and the key images selected by our approach proved to be good enough for the navigation. Our navigation algorithm is based on the idea that the key images around the immediate neighborhood of the robot have more matches than others. The bar graphs in Figs. 6.6, 6.11 and 6.12 confirm this idea. The adjacent key images that have maximum common lines give the initial location in the map. Based upon line matching results, the key images are switched automatically and the appropriate rotational velocity is set that allows the robot to follow the learned path. IBVS has been able to keep the error within small bounds. The robot did not exactly follow the learned path because neither 3D information nor any 3D motion estimation to correct the pose was used, as this is not our objective of the navigation to be accurate, but to be successful and robust. The other reason is also due to approximation of the path by straight lines. However, neglecting 3D information also results in some limitations like more lateral deviation especially after sharp turnings. Nevertheless, based on 2D information only, a useful navigation could be performed in the corridors and inside the room as visual servoing is robust enough to handle such errors.

Comparing with point based method [DSRC11, ŠRDC09], our method performs better especially in low textured environment as in turnings of corridor and in-presence of motion blur due to rapid camera motion as shown in Fig. 6.17, where reliable point based features cannot be detected, resulting in failure in tracking and 3D reconstruction (without using external informational from sensors like IMU). Especially in indoor environment, it is very difficult to track feature points reliably over a longer sequence. This method based on line segments also performs better in the environments like those in Figs. 6.13 and 6.17 in comparison to MI-based navigation (Chapter 5) because line segments are abundant in the structured indoor environment and are also more resilient to motion blur and partial occlusions, whereas MI-based navigation is sensitive to changes in the scene due to new objects and occlusions and performs poorly when there is less or similar texture especially in turnings of the corridor. However, our framework also has some limitations that are mainly due to the line matching algorithm, which is not still a mature field in computer vision unlike points, especially in the cases where there are very few line segments detected in the images. Initial localization might produce false results when there are few matches (say less than 10) by using just two view matching. In such cases, 3 view matching can be used for verification to select the two nodes in the map. However, most of these problems can be greatly avoided by selecting proper trajectory during the mapping. The other problem that might be encountered in our framework is a singularity of J_a (of (5.23)). During all our experiments, singularity condition never occurred. The smallest value of J_a encountered during our all experiments was -0.0018. In most cases, the value of J_a is always greater than 1.

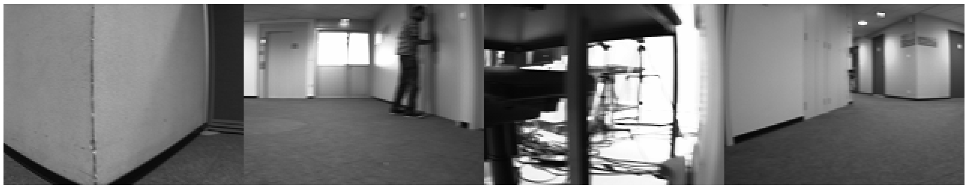


Figure 6.17 – Some cases where point based methods [DSRC11, ŠRDC09] failed and our approach succeeds.

6.4 Conclusions

We have presented a method for indoor qualitative mapping and navigation based on a topological representation of the environment using only line segments extracted from a perspective camera. Our navigation is exclusively based on 2D image measurement without relying on any 3D reconstruction process as in most existing literature. This is possible due to a topological representation of the environment and the use of image based visual servoing for motion control. Using line segments as features makes navigation possible despite some level of occlusions and blur in the image. Unlike previous method using MI that uses entire image, our method is insensitive to changes outside of the tracked/matched lines. Besides that, it is very hard to discriminate between two places from MI, which have similar histograms. Such cases are better handled by this method because of the robustness of line matching algorithm. Unlike point features, line segments are better tracked/ matched over the longer sequences especially in the indoor environment. The experiments showed that the method is able to handle moderate changes in lighting conditions and new objects in the environment. Difficult situations include featureless areas like smooth/texture-less walls (especially for sharp turnings), photometric variations like strong shadows, rapid and sharp turnings. More elaborate image processing like preprocessing the image and and/or robot control strategies like filtering rotational velocity can address these issues.

CHAPTER 7

Combining Line Segments and Points for Appearance-Based Indoor Navigation

THIS chapter presents image-based navigation from an image memory using a combination of line segments and feature points. The method presented here is the extension of the work of Chapter 6. Based upon line segments matching and tracking of points, an image memory is built in off-line mapping phase. During navigation, the common line segments and feature points between the current acquired image and the nearby key images is exploited to switch the key images and to derive a control law for computing the rotational velocity. Using our approach, real-time navigation has been performed in real indoor environment with a Pioneer 3-DX equipped with an on-board perspective camera. We show that the combination of points and lines increases the number of features that helps in robust and successful navigation especially in those regions where few points or lines can be detected and tracked/matched.

The work of this chapter has been submitted to ICRA-2017 [BRGCew] for review.

7.1 Advantages of Combining Multiple Features

Most of the techniques employed in outdoor environments typically results in a significant performance drop when applied to indoor scenarios because of windows, wiry structures, reflections and repetitions, as well as limited texture in indoor scenarios, which causes standard procedures based on image descriptors to poorly perform indoors [MW15]. However, in our previous chapter we showed that line segments are actually good features for indoor navigation. Still, there are some limitations mainly due to the line matching/tracking algorithms, which are still not a mature field in computer vision unlike points, especially when very few line segments can be detected in the images. Therefore, in this chapter we extend the navigation framework proposed in Chapter 6 to combine line segments with point features for increasing robustness in terms of its application to the wide range of indoor scenarios with smooth motion.

Our Contribution

Our main contribution is a complete method for indoor navigation like in previous chapters. Up-to our knowledge, combining general line segments with features points for the navigation has never been done before in the literature. This combination is particularly effective in those cases where there are few lines detected and matched, especially during sharp turnings and changes in the illumination conditions. Using points with lines will increase the number of features at those region, and eventually help in better motion control and hence more robust navigation.

7.2 Our Method

Edge Drawing Lines (EDLines) detector [AT11] and the line matching method proposed by [ZK13] have been used to detect and to match the line segments, respectively. SURF points [BETVG08] and the corners detected by FAST corner detector [RPD10] have been considered as point features. Wide baseline matching of the points is performed only during the initial localization. During the navigation, the points (only FAST corners have been used for successive localization) from the key images are tracked using modified KLT (Kanade-Lucas-Tomasi Feature Tracker) algorithm as presented in [DSRC11]. Trifocal tensor with RANSAC (ref. Sect. 2.1.3.4) has been used to remove outliers for the line segments matching similar to the process described in our previous chapter. The outliers of point matching/tracking are removed by using 5 point algorithm with RANSAC [Nis04].

As in Chapter 6, two view correspondences are used in initial localization, switching of key images and generating three view correspondences. Three view correspondences are used in mapping, switching of key images and motion control. For generating three view correspondences, the current key image and the two most recently acquired images are used during the mapping, whereas, the two key images and the currently acquired image are used during the navigation. When obtaining the three view correspondences with line segments, only the correspondences between the first two images are used to match with the third image in order to reduce the cost of matching. In our method, we have used only those line segments and points that are geometrically consistent (2 view geometry for points and 3 view geometry for lines) and pass inlier test using RANSAC. Therefore, we have confidence over the lines and points that have been used that allows to use number of correspondences as weighting metrics.

7.2.1 Key Images Selection

Let I_c be the most recently acquired image and I_k be the most recent key image. The line segments and corners detected in I_k are tracked over I_c. For tracking of lines, we have used the method proposed in previous chapter. For tracking of points, the approach used in [DSRC11] has been used. If $n_L(I_k, I_{c-1}, I_c)$ and $n_P(I_k, I_c)$ denotes the numbers of lines and points tracked after outliers rejection and ε is a positive integer, the new key image is selected when

$$\boxed{n_L(I_k, I_{c-1}, I_c) + n_P(I_k, I_c) < \varepsilon.}$$

In this case I_{c-1} will be new I_k. The neighboring key images share some common line segments and feature points as shown in Fig. 7.1, which makes it possible to consider multiple key images in a neighborhood for defining the heading angle of the robot.

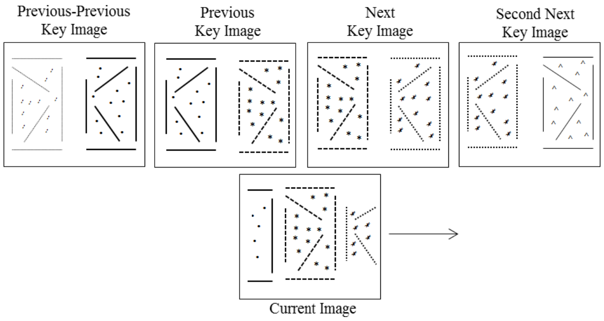


Figure 7.1 – The map consists of key images. Adjacent key images share some line segments and feature points with the current image. These corresponding features with the current acquired image are used for switching of key images and motion control.

7.2.2 Navigation in the Map

7.2.2.1 Qualitative Localization

Initial Localization in the map The navigation starts with the initial localization which allows determining the initial position of the robot in the map. The first image acquired (I_a) is compared with all the images in the database based upon line segment and points matching. If I_k denotes key image, $n(...)$ is the total number of lines and points matched between the images, I_P is previous key image and I_N is next key image, I_P and I_N are selected as:

$$\boxed{\begin{array}{c} I_{k_i} = \arg\max_{I_k} \{n(I_a, I_{k_1}), n(I_a, I_{k_2}), ..., n(I_a, I_{k_n})\} \\ I_{k_j} = \arg\max_{I_k} \{n(I_a, I_{k_{i-1}}), n(I_a, I_{k_{i+1}})\} \\ \\ I_P = I_{k_j} \text{ and } \quad I_N = I_{k_i} \text{if } i > j \\ I_P = I_{k_i} \text{ and } \quad I_N = I_{k_j} \text{if } i < j \end{array}}$$

Successive Localization (Key Images Switching) After initial localization in the map, further localizations can be done exploiting the topological relationship between the key images. This can be done by comparing with only few adjacent images in the database. The previous key image I_P, the next key image I_N and the second next key image I_{NN} are compared with the current acquired image I_a. Let $n_L(...)$ be the total number of lines matched and $n_P(...)$ be total number of points tracked between the images. Then switching of key images is done when at least one of the following criteria is fulfilled for two consecutive acquired images I_a and I_{a+1}

$$\boxed{\begin{array}{c} (n_L(I_a,I_N,I_{NN})+n_P(I_a,I_N,I_{NN})) > (n_L(I_P,I_a,I_N)+n_P(I_P,I_a,I_N)) \text{ or} \\ (n_L(I_a,I_{NN})+n_P(I_a,I_{NN})) > (n_L(I_a,I_N)+n_P(I_a,I_N)) \text{ and} \\ (n_L(I_a,I_{NN})+n_P(I_a,I_{NN})) > (n_L(I_p,I_a)+n_P(I_p,I_a)). \end{array}}$$

The first criterion is based on the result of three view matching between the images inside the brackets, whereas the second criterion is based on two view matching of images. The second criterion is essentially useful when there are no three view matches or very few number of three view correspondences that may occur in the rapid turnings.

7.2.2.2 Motion Control

As per the previous methods proposed in this thesis, for navigation, the robot is not required to accurately reach each key image of the path, or to accurately follow the learned path. We follow the same strategy as before with **s** (ref. (4.10)) a combination of lines and points.

For line segments, we use the abscissa of the centroid of the points of intersection of the n matched lines and their respective normal from the origin (Fig. 6.4 (right)) as in our previous chapter. For the points, we have used x-coordinate of the centroid of the m tracked points as in [DSRC11].

The interaction matrix for line segments has been derived in the previous chapter and it takes the expression

$$J_{vL} \simeq 0 \text{ and } J_{\omega L} \simeq \frac{1}{n}\sum_{i=1}^{n}(\cos^2\theta_i - \rho_i^2\cos(2\theta_i)), \tag{7.1}$$

where (ρ_i, θ_i) are the line parameters (as shown in Fig. 6.4 (right)) of the matched lines in I_a, and J_{vL} and $J_{\omega L}$ are the Jacobians for translational velocity and rotational velocity respectively.

From [CH06] and (4.6), the interaction matrix $(\mathbf{L_x})$ of point $(x\,,y)$ that links its displacement w.r.t. to robot velocity $(v_r\,,\,\omega_r)$ is

$$\mathbf{L_x} = \left[\begin{array}{cc} \frac{x}{Z} & -\left(\frac{\delta}{Z}+1+x^2\right) \end{array}\right], \tag{7.2}$$

where x and y are normalized image coordinates and Z is the depth of the point.

Using the x-coordinate of the centroid of m tracked points as feature, the Jacobians for translational velocity (J_{vP}) and rotational velocity $(J_{\omega P})$ are given as

$$\begin{aligned} J_{vP} &= \frac{1}{m}\sum_{j=1}^{m}\left(\frac{x_j}{Z_j}\right), \\ J_{\omega P} &= \frac{1}{m}\sum_{j=1}^{m}\left(\frac{\delta}{Z_j}+1+x_j^2\right). \end{aligned} \tag{7.3}$$

Neglecting δ w.r.t. Z (depth of the point), Z w.r.t x i.e. $Z \gg x$, and assuming the camera optical axis is orthogonal to the axis of robot rotation and that the centroid stays near the image plane center, we obtain

$$J_{vP} \simeq 0 \text{ and } J_{\omega P} \simeq \frac{1}{m}\sum_{j=1}^{m}(1+x_j^2),\ \delta \ll Z. \tag{7.4}$$

Since we only control ω_r, only one feature derived from all line segments and points is sufficient. Therefore, instead of using points and lines as separate features to derive the rotational velocity, we combine them such that lines and points are treated equally, and we can take advantage of all the lines and points matched/tracked. We can combine them either taking weighted average or using linear least square.

Using Weighted Average

We recall that let X_a, X_N and X_{NN} be the abscissa of the centroid of n matched lines of I_a, I_N and I_{NN} respectively, h_1 and h_2 be positive weights such that $h_1+h_2=1$, λ_L be the positive gain, and ε be a small value to prevent division by 0. The expression for the rotational velocity using line segments only (ω_{rL}) is can be obtained from (6.13) as

$$\omega_{rL} = -\frac{\lambda_L}{J_{\omega L}\pm\varepsilon}(h_1(X_a-X_N)+h_2(X_a-X_{NN})). \tag{7.5}$$

Let x_a, x_n and x_{nn} be the x-coordinate of the centroid of tracked points of I_a, I_N and I_{NN} respectively, λ_P be a positive gain, and g_1 and g_2 positive weights such that $g_1+g_2=1$. The expression for the rotational velocity using only point features can be computed similarly to the case of lines as

$$\omega_{rP} = -\frac{\lambda_P}{J_{\omega P}}(g_1(x_a-x_N)+g_2(x_a-x_{NN})). \tag{7.6}$$

The final expression for the rotational velocity (ω_r) is then obtained by taking the weighted average of (7.5) and (7.6) as

$$\omega_r = \frac{n\,\omega_{rL}+m\,\omega_{rP}}{n+m}. \tag{7.7}$$

If points cannot be tracked which may occur in practice, (7.7) considers only line segments, and vice versa. This approach is also modular and can be easily used to combine different geometric feature based methods.

Using Linear Least Squares

Let X_a, X_N and X_{NN} be the abscissa of the centroid of the points of intersection of n matched lines and their respective normal from the origin of I_a, I_N and I_{NN} respectively. Then, the error term for the case of line segments is $X_a - X_N$.

Let x_a, x_n and x_{nn} be the x-coordinate of the centroid of tracked points of I_a, I_N and I_{NN} respectively. Then, the error term for the case of points is $x_a - x_n$.

Combining lines and points together, the expression for the rotational velocity can be computed from (5.17), (6.11) and (7.4) as

$$\omega_r = \begin{bmatrix} J_{\omega L} \\ J_{\omega P} \end{bmatrix}^{+} \left(\lambda \begin{bmatrix} (X_a - X_N) \\ (x_a - x_n) \end{bmatrix} + \begin{bmatrix} J_{vL} \\ J_{vP} \end{bmatrix} v_r \right). \tag{7.8}$$

Since J_{vL} and J_{vP} are close to zero (from (6.11) and (7.4)), after simplification, (7.8) becomes

$$\omega_r = -\frac{\lambda \left(J_{\omega L}(X_a - X_N) + J_{\omega P}(x_a - x_n) \right)}{J_{\omega L}^2 + J_{\omega P}^2}. \tag{7.9}$$

In order to smooth the rapid steering actions when switching between frames, a feed-forward command is also added to ω_r. The calculation of the feed-forward term is based on the difference of the centroids between the shared lines and points of I_a with I_N and I_{NN}. The final expression for the rotational velocity can be computed as

$$\omega_r = -\frac{\lambda \left(J_{\omega L}(g_1(X_a - X_N) + g_2(X_a - X_{NN})) + J_{\omega P}(g_1(x_a - x_n) + g_2(x_a - x_{nn})) \right)}{J_{\omega L}^2 + J_{\omega P}^2}, \tag{7.10}$$

where g_1 and g_2 are positive weights such that $g_1 + g_2 = 1$. This is more general approach than the previous one. With this approach, we can combine geometric feature based methods and non geometric feature based methods (like mutual information) also.

Thus, our complete framework uses only 2D information obtained from line segments and feature points. From this information, we derive the required rotational velocity using a *IBVS* control law, which makes the robot to follow the learned path successfully without any need of 3D or accurate mapping and localization. The comparison between (7.7) and (7.10) is discussed in Sect. 7.3.3.

Stability Analysis

Let $J_{\omega L}$ and $J_{\omega P}$ be the true Jacobians and $\hat{J}_{\omega L}$ and $\hat{J}_{\omega P}$ be the approximated Jacobians related to ω_r for lines (that with subscript L) and points (that with subscript P) features. The global asymptotic stability can be guaranteed with a feature derived from all line segments is shown in Sect. 6.2.3.2. From (7.3) for points only, we have

$$J_{\omega P} = \frac{1}{m}\sum_{j=1}^{m}\left(\frac{\delta}{Z_j}+1+x_j^2\right) > 0 \text{ and } \hat{J_{\omega P}} \simeq \frac{1}{m}\sum_{j=1}^{m}(1+x_j^2) > 0. \tag{7.11}$$

Therefore,

$$\frac{J_{\omega P}}{\hat{J_{\omega P}}} = 1 + \frac{\frac{1}{m}\sum\limits_{j=1}^{m}\left(\frac{\delta}{Z_j}\right)}{\frac{1}{m}\sum\limits_{j=1}^{m}\left(1+x_j^2\right)} > 0, \tag{7.12}$$

since $\delta \ll Z_j$ and $\frac{1}{m}\sum\limits_{j=1}^{m}\left(1+x_j^2\right)$ is close to 1. From (7.4), we have Jacobian related to v_r is close to zero. Hence, with (7.12), the global asymptotic stability is guaranteed as inferred by (4.19) for a feature derived from all the matched points.

The case of weighted average corresponds to the single-feature case, for which global stability can be ensured. For the linear least square, we have

$$\begin{bmatrix} J_{\omega L} \\ J_{\omega P} \end{bmatrix} \begin{bmatrix} \hat{J_{\omega L}} \\ \hat{J_{\omega P}} \end{bmatrix}^{+} = \frac{1}{\hat{J_{\omega L}^2}+\hat{J_{\omega P}^2}} \begin{bmatrix} J_{\omega L}\hat{J_{\omega L}} & J_{\omega L}\hat{J_{\omega P}} \\ J_{\omega P}\hat{J_{\omega L}} & J_{\omega P}\hat{J_{\omega P}} \end{bmatrix} \geq 0, \tag{7.13}$$

which is positive semi-definite because one of its eigen value is 0 and other is $J_{\omega L}\hat{J_{\omega L}} + J_{\omega P}\hat{J_{\omega P}} > 0$. From (7.1) and (7.4), we have Jacobians related to v_r is close to zero. Therefore, with (7.13) linear least square is stable only not asymptotically stable.

7.3 Results and Discussions

The mapping was performed off-line, whereas the navigation experiment was performed on-line at 5 Hz. In all our experiments, we have used $h_1 = g_1 = 0.7$, $h_2 = g_2 = 0.3$, and $\lambda_L = \lambda_P = \lambda = 1$. As usual, the trajectories presented below are obtained from the odometry of the Pioneer Robot. The videos of the experiments are available in [1]

7.3.1 Mapping

The number of images acquired and selected key images in different experiments are presented in Table 7.1, where $n(Acq.)$ represents the total number of acquired images and $n(Ref.)$ represents the total number of key images selected. The trajectory obtained from the odometry is shown by a red curve in Figs. 7.2, 7.4, 7.5, and 7.6, where the red symbol $*$

[1] http://www.irisa.fr/lagadic/team/Suman.Bista.html#videos.

represents the location of the key images. There are more key images over a small distance in case of quick displacements of features like during turnings or when line segments and points cannot be successively matched/tracked over the sequence because of, e.g., changes in the illumination.

	Experiment	$n(Acq.)$	$n(Ref.)$
I	Inside Robotics room	1260	19
II	Room-Corridor-Robotics room	4555	105
III	Robotics Room-Corridor	4100	52
IV	Corridor Out-In	10297	132
V	Corridor In1	7021	140
VI	Corridor In2	6530	122

Table 7.1 – Table Showing the number of acquired images and selected key images in the different experimental scenarios.

7.3.2 Navigation

The robot was placed inside the mapped environment with the camera facing towards the mapped direction (Initial position shown by green dot). The forward velocity was set to 0.15 m/s and reduced to 0.08m/s when turning, whereas the rotational velocity was controlled by the navigation algorithm using (7.10). Such turnings are automatically detected by observing the commanded rotational velocity. During navigation, the robot has been able to follow the learned trajectory as shown by the blue curve in Fig. 7.2, and Figs. 5.6-7.7, with the automatic switching of the key images.

7.3.2.1 Navigation inside the robotics room

The navigation has been performed with some changes in environment, for instance the chair and table have been added to the environment after the mapping phase. Besides, the chair was moving from one point to another and a person was walking during the navigation. These changes can be clearly seen in Fig. 7.2 (middle and right). Fig. 7.2 (left) shows the successful navigation of the Pioneer in the environment despite the various changes. After turning, the majority of lines detected and matched belongs to the floor only. But thanks to the points, the robot was able to obtain features from the walls and from other objects. With more and better distribution of obtained features from points and lines, the motion was smoother especially after turning, which was not the case using line segments only (refer Fig 7.3).

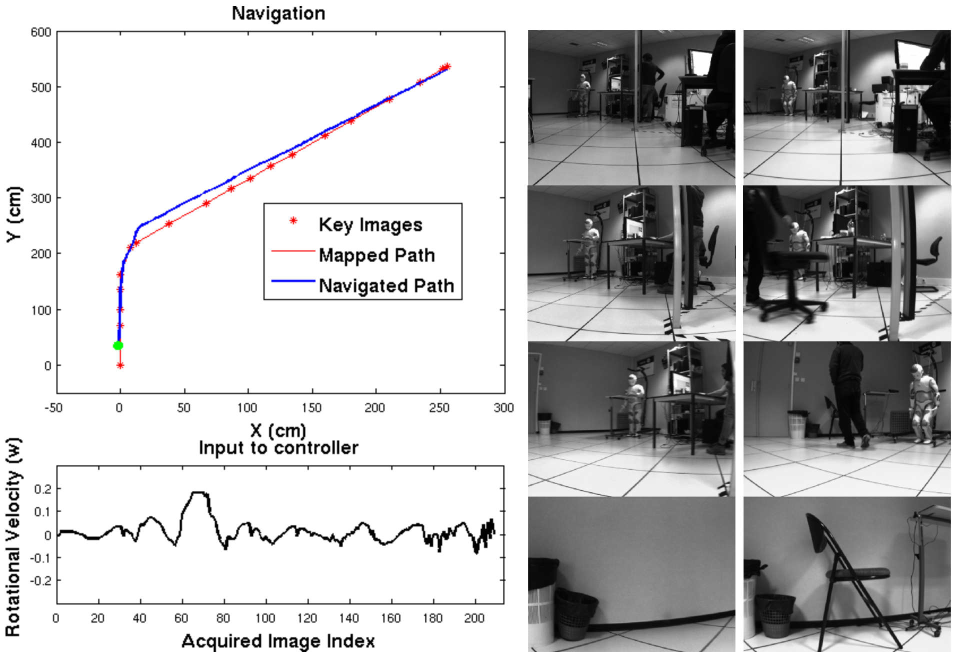


Figure 7.2 – Navigation inside the robotics room (left), some key images (middle), and some changes in environment during navigation (right).

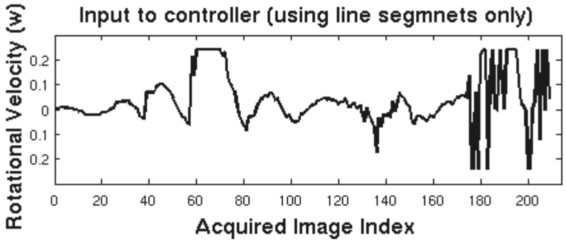


Figure 7.3 – Rotational velocity input with line segments only in the same environment as in Fig. 7.2

7.3.2.2 Navigation from a room to another room via a corridor

Fig. 7.4 shows the navigation between rooms via a corridor. The robot followed the learned path of 22m with turnings whenever it was required. Right angle turning, especially the second one (in the corridor), was a challenging task. This turning was very difficult using line segments only because of few lines matched and illumination conditions, which were not always sufficient to get a large enough rotational velocity for this sharp turning. However, together with the points, the number of features in this region increased significantly, which allowed the system to be successful during the sharp turning. The lateral drift was within 5 cm from the mapped position, thus confirming the accuracy of the visual servoing control law.

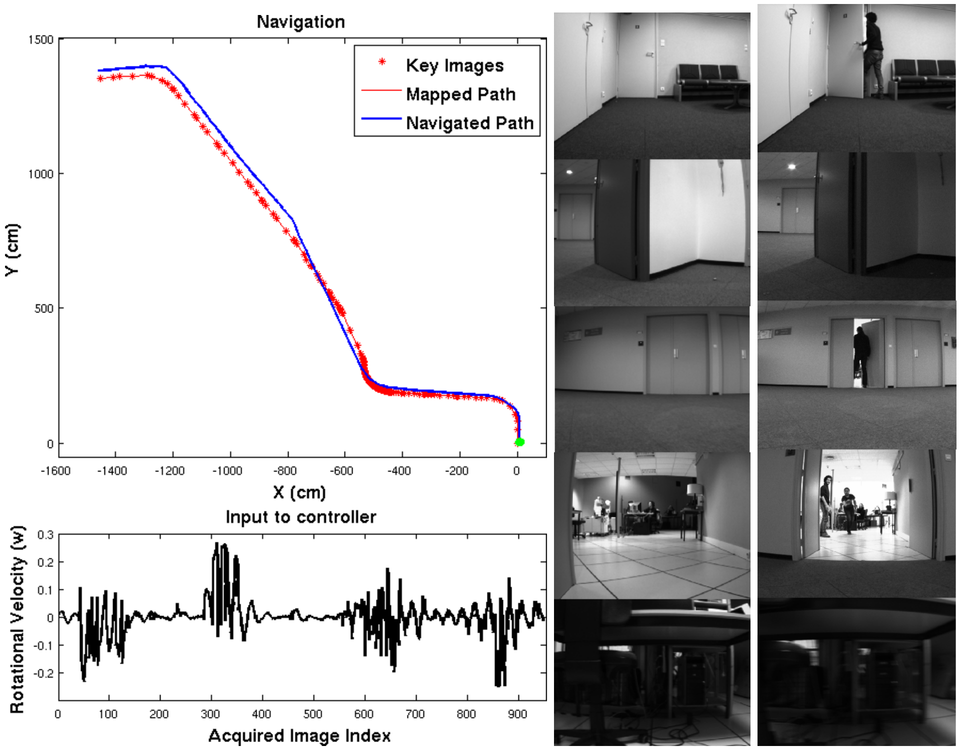


Figure 7.4 – Navigation between the rooms via corridor (left), some key images (middle), and some images during navigation (right).

7.3.2.3 Navigation from a room to the corridor

In this case, we performed the navigation in the 23m path that starts from the robotics room and navigates into the corridor. The path consists of multiple turns and different levels of illumination as shown in Fig. 7.5. Even in presence of moving people and objects, the robot was able to follow the learned trajectory. With the addition of point features, the motion was relatively smooth because of the increased number and better distribution of features to compute the rotational velocity.

7.3.2.4 Navigation in a corridor (Out-In)

In this case, we performed the navigation of 47m in a corridor (mapped length is 51m). The beginning of the navigation zone consists of large windows that allow seeing outdoor. So, the robot undergoes large changes in the illumination in the mapped path as we enter inside the corridor. Also, there are two small inclined planes in the path. Even in presence of changing lightning condition (cloudy when mapped and sunny during navigation) and walking people, the robot was able to follow the learned trajectory as shown in Fig. 7.6. Especially due to illumination change, the number of lines matched decreased. However, the addition of point features allowed for a smooth navigation. For this experiment we

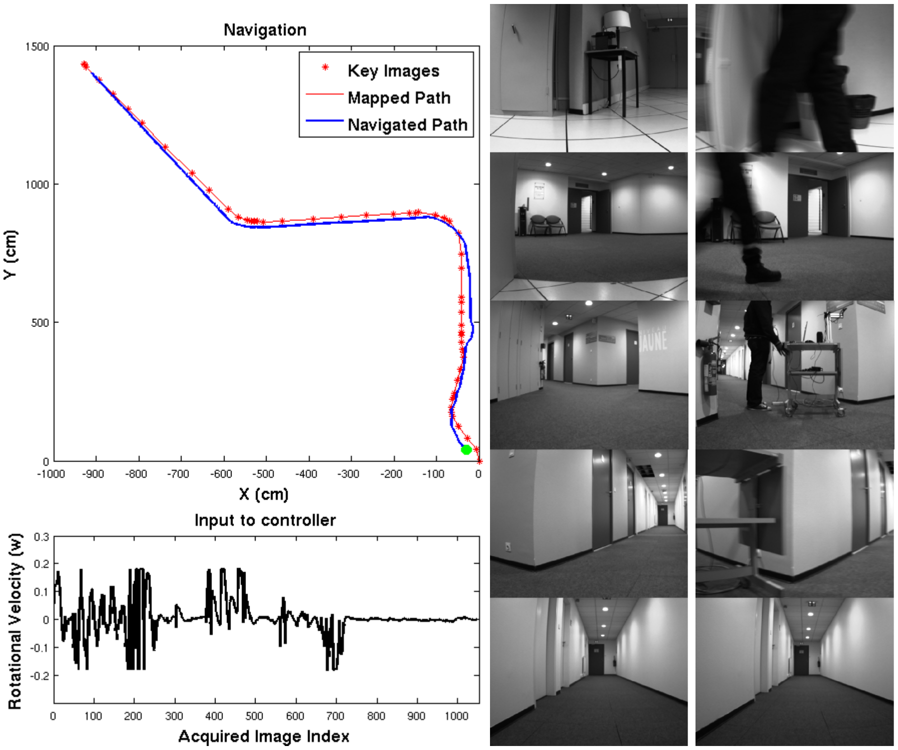


Figure 7.5 – Navigation in between the room and the corridor (left), some key images (middle), and some images during navigation (right).

started the robot 4m from the initial point of the mapped location. The robot successfully navigated the remaining path by automatically selecting the initial key images, thanks to the initial localization.

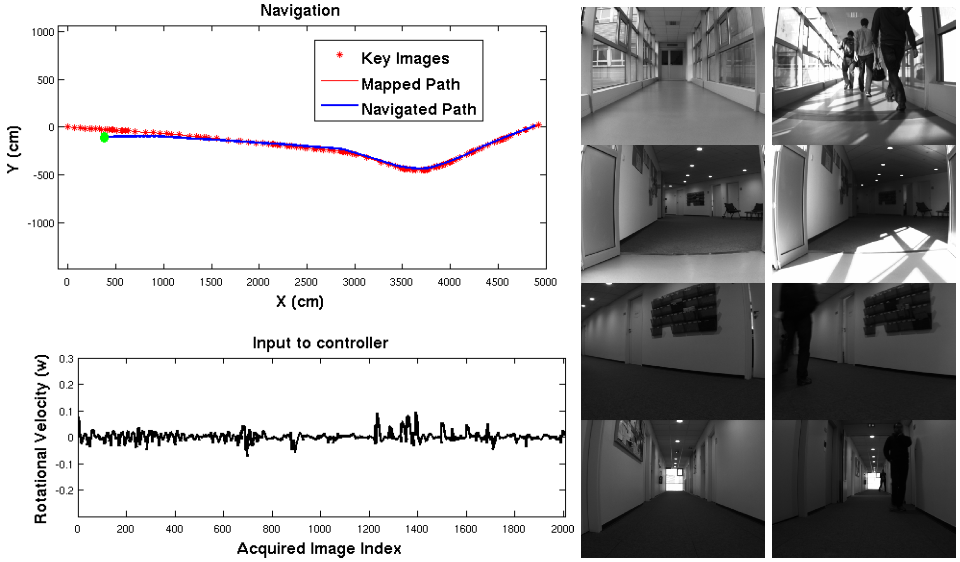


Figure 7.6 – Navigation in the corridor Out-In (left), some key images (middle), and some images during navigation (right).

7.3.2.5 Navigation in indoor corridor (In1, In2)

Here, we performed navigation in two corridors that are completely indoor. The corridors have shiny floor and walls are mostly white. The illumination in first corridor (In1, length 32m) totally depends upon the bulbs on the top of the ceilings where as in second corridor (In2, length 34m) there is also light from outside. Navigation in both cases were difficult using line segments only because of the type of surface of walls very few line segments can only be detected and matched in some regions and in turnings. With lines only as in previous chapter or points only as in [DSRC11], navigation was not successful. Using points in addition to lines, more number of features are present which makes it possible for successful navigation as shown in Fig. 7.7.

7.3.3 Discussion

The presented results show the viability of our approach in different scenarios and constraints. The robot has been able to follow autonomously the learned path from the start position. Our framework does not depend upon any particular type of line segment or points but it is just based on generic lines or points that are detected and matched/tracked from key images. The key images selected by our approach proved to be good enough for the navigation. Based upon the line matching and tracking of the points from neighboring key frames, the key images are switched automatically and an appropriate rotational velocity can be computed for allowing the robot to follow the learned path. The IBVS law has been able to keep the error within small bounds. The deviation from the learned path is within the range of 5 cm. The rotational velocities obtained from combining points and lines with weighted average and least square are almost same. One of such instance is presented in Fig. 7.8, where the blue curve is with weighted average and green one with linear least square.

Our framework takes advantages from both lines and points. Line segments are abundant in a structured indoor environment, and they are also more resilient to motion blur and partial occlusions. Even if the performance of point features can degrade in the indoor environments, they can still be tracked locally to some extent, which is sufficient to supplement the line segments. This then helps in better switching the key images and in computing the rotational velocity. The robust sharp turning, smooth motion of the robots, and successful navigation in wide range of environment are the consequences of using multiple features. Besides that, the point tracking and the line matching can be operated in parallel without any significant increase in the computational time. We still manage to operate at 5Hz with the improved performance as in the case of line segments only.

However, our framework has also some limitations that are mainly due to the line matching and the point tracking algorithm especially in the cases where very few line segments are detected in the images and the tracking of points does not perform well. During all

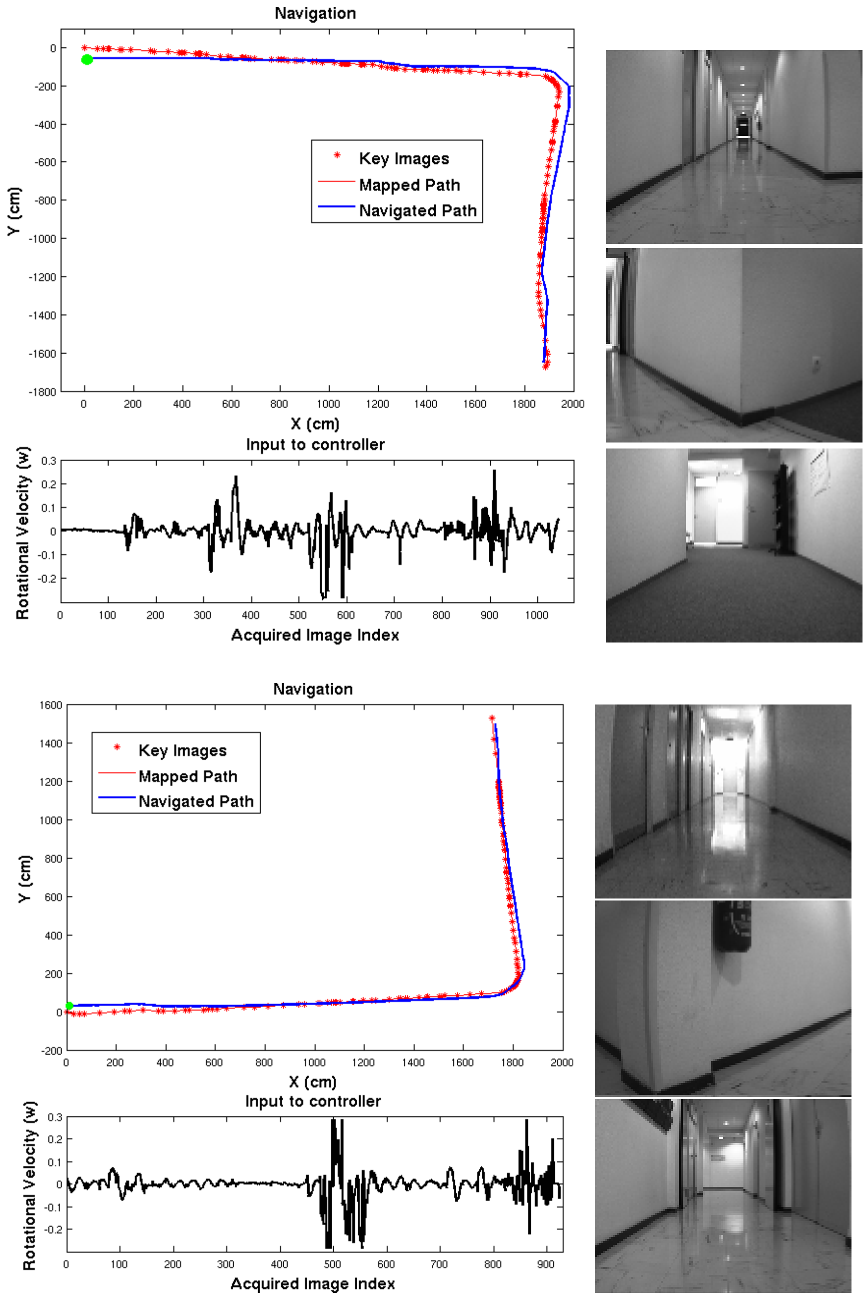


Figure 7.7 – Navigation in the corridor : In1 (up) and In2 (down).

our experiments, singularities in $J_{\omega L}$ (in (5.23)) never occurred. Initial localization might produce false results when there are few matches that make the geometrical verification of matched points/lines not possible. However, most of above problems can be greatly avoided by selecting a proper trajectory during the mapping. Initial localization might produce false results when there are few matches that make the geometrical verification of matched points/lines not possible. However, most of above problems can be greatly avoided by selecting a proper trajectory during the mapping.

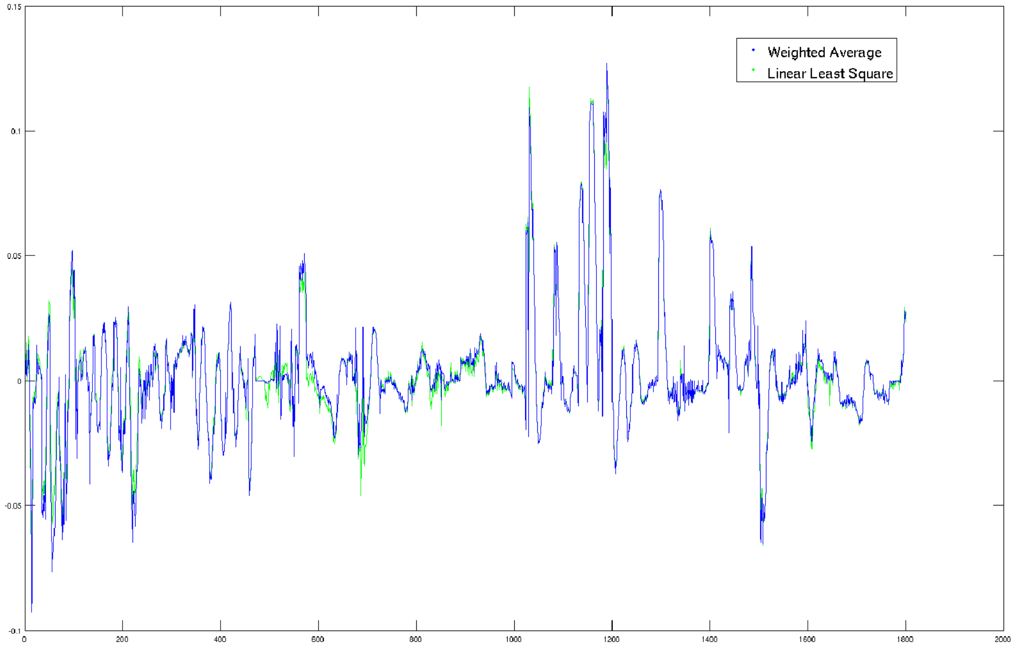


Figure 7.8 – Rotational Velocities using two approaches.

7.4 Conclusions

We have presented, in this chapter, a method for indoor qualitative mapping and navigation based on image memory using a combination of line segments and points, by expanding our previous work as per the other strategies presented in this thesis. Our navigation is exclusively based on 2D image information without relying on any 3D reconstruction or pose estimation, and also without accurately tracking the trajectory used in the learning phase. Combination of points and line features increases the number of features in the image which results in a more robust navigation and smoother control. Increased number of features plays an important role especially during sharp turnings where less features can be detected, which changes rapidly also. We showed a successful navigation in different indoor scenarios while being robust to some level of occlusions and blur in the image, moderate changes in lighting conditions and presence of new objects in the environment. Difficult situations include featureless areas like smooth/texture-less walls. Apart for the initial localization, we have used only the corners detected by FAST algorithm for the navigation. However, we can easily incorporate other interest point detectors like SIFT/-SURF/BRISK in our framework so as to increase its robustness. Unlike the method using Mutual Information that uses entire image, this method is insensitive to changes outside of the tracked/matched lines and points. Hence, this method handles occlusions and changes in the environment better than using entire image too. Besides, more precise image processing like pre-processing images and/or robot control strategies, like using vehicle kinematics and filtering the velocity, could be incorporated to overcome uncertainties in the navigation.

Part III

Summary

CHAPTER **8**

Conclusions and Future Work

8.1 Summary and Contributions

We have presented a complete method for indoor qualitative mapping and navigation based on appearance. Our navigation is exclusively based on 2D image measurements without relying on any 3D reconstruction process as in most existing literature. This is possible due to a topological representation of the environment and the use of IBVS for motion control. Topological representation avoids the need of accurate metric maps, which are not easy to create (and sometimes not possible to create due to lack of sufficient features to obtain the desired accuracy) and are computationally expensive. The map is as simple as just a set of reference images that represents the learned path. Using IBVS reduces computational delay, eliminates the necessity for image interpretation, and errors due to camera modeling and calibration. The use of vision and control in the closed loop via IBVS makes it possible for the robot to follow the learned path keeping error under the certain bounds from the mapped location even without 3D information. Our navigation method includes mapping, localization and motion control. The mapping process is done offline, where the operator drives the robot manually in the environment where it has to navigate capturing the images of the path. From this learning sequence, the image memory is created by automatically selecting the reference/key images. Navigation process is online, where the robot initially localizes itself in the map at first. Then, it continuously performs successive localizations to select the appropriate reference images and use multiple reference images for control so that robot follows the learned path. Our method is applicable to both local and global features. First, we have used the entire image without any feature extraction and matching, where shared entropy between the current view and near images is exploited for the mutual information-based navigation. This method works well even for changes in the illumination but it performs poorly when there are significant occlusions, and the environment lacks sufficient texture or same texture. Using local features avoids this problem to a greater extent. We have used first line segments alone, which are present in the numerous amount in the structured indoor environment. Using line segments alone makes it possible for mobile robots like Pioneer to navigate just relying upon the information from 2D lines, unlike in [ŠRDC09, DSRC11], where point-based features are used along with local 3D reconstruction for outdoor navigation. The navigation works well for the scenarios where line segments are present in sufficient number. During some turnings and in some corridors there are few line segments matches that are either due to the presence of few lines or the poor performance of line matching algorithms due to illumination conditions and similar textures, where our method based on line segments performs poorly. In other to improve the performance, we combined lines segments with points. This combination increases the number of features, which ultimately increases its robustness and range of environment for its application despite the performance of points and its tracker decreases significantly in indoor scenarios. The combination of general line segments and points for indoor navigation has rarely been done in the literature. Our combination scheme of multiple features is equally applicable to other types of features which we show in following sections.

8.2 Open Issues and Future Perspectives

The results of this thesis, although encouraging, also show a number of limitations that affect our approach. First one is related to feature detection and matching. The performance of our method depends upon the performance of feature matcher/tracker. If few points and/or lines are detected, our local feature based method performs poorly. In other hand, with less-textured surface and similar textured surface, mutual information based navigation performs badly or fails. Also, the mutual information-based method is poor in handling occlusions especially for switching of reference images. However, most of above problems can be avoided to some extent by selecting a proper trajectory during the mapping. The other solution is to make use of multiple features, which can be easily combined similar to (7.7) and (7.8). In (7.7), instead of using weighted average on the number of instances (number of instances is not applicable for mutual information), confidence value can be used to weight these contributions from local and global features. In (7.8), the combination can be achieved simply by stacking Jacobians and errors in their respective places. The contributions from different features can be performed in parallel without compromising the performance.

Second, the presented method is concerned only with a goal-directed behavior without considering obstacle avoidance. Thus, in the navigation experiments, we assume that other moving objects will adopt collision-free trajectories. However, the obstacle avoidance framework can easily be incorporated in our method using laser similar to [CC13]. We can use obstacle avoidance module to control forward velocity (v) and to maintain the visibility of features while avoiding obstacles, camera pan angle (φ) also has to be controlled as shown in Fig. 2.16(right). Hence, in this case, our visual task is

$$\dot{\mathbf{s}} = \boldsymbol{J}_v v + \boldsymbol{J}_\omega \omega + \boldsymbol{J}_{\dot{\varphi}} \dot{\varphi}, \tag{8.1}$$

where ω is the rotation velocity and $\boldsymbol{J}_v$, $\boldsymbol{J}_\omega$ and $\boldsymbol{J}_{\dot{\varphi}}$ are the Jacobians related to v, ω and $\dot{\varphi}$ respectively. In our previous methods, from (5.21), (6.11), (7.1) and (7.4) we can safely assume

$$\boldsymbol{J}_v \simeq \mathbf{0}. \tag{8.2}$$

Let ω_r be the rotational velocity obtained from our previous method as in (5.23), (6.13), (7.7), and (7.10). Then we can control $(v, \omega, \dot{\varphi})$ using (8.1), considering different scenarios as follows:

In safe Context (No obstacle)

$v = v_s.$	v_s is safe translational velocity.
$\omega = \omega_r + \lambda_\varphi \boldsymbol{J}_\omega^+ \boldsymbol{J}_{\dot{\varphi}} \varphi.$	λ_φ is a gain.
$\dot{\varphi} = -\lambda_\varphi \varphi.$	This reduces pan angle to zero.

In unsafe Context (Obstacle Very Near)

$v = 0.$	
$\omega = \lambda \alpha.$	α is calculated from obstacle distance of the safest path. λ is a gain.
$\dot{\phi} = \boldsymbol{J}_{\dot{\varphi}}^+ \boldsymbol{J}_w (\omega_r - \lambda \alpha).$	This maintains feature visibility.

Intermediate Situations

If H is the risk function of best tentacle i.e. the curvature robot has to follow to avoid obstacle (see Fig. 8.1), then we have

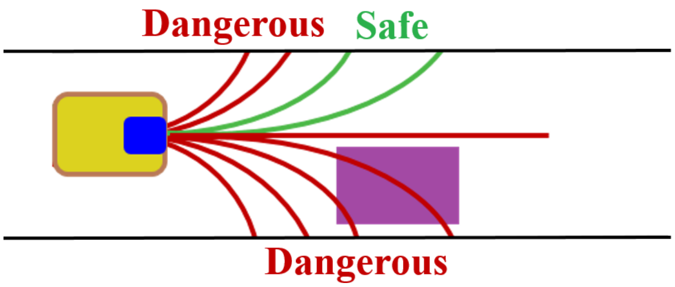


Figure 8.1 – Tentacles for obstacle avoidance.

$v = (1 - H) v_s.$	$H = 0$ means no obstacle or obstacle is far away.
$\omega = (1 - H)(\omega_r + \lambda_\varphi \boldsymbol{J}_\omega^+ \boldsymbol{J}_{\dot{\varphi}} \varphi) + H \lambda \alpha.$	$H = 1$ means obstacle is very close.
$\dot{\phi} = H \boldsymbol{J}_{\dot{\varphi}}^+ \boldsymbol{J}_w (\omega_r - \lambda \alpha) - (1 - H) \lambda_\varphi \varphi.$	$H = [0, 1]$ means obstacle is in the range to be considered.

The details about tentacles and collision avoidance are presented in [CC13].

Finally, the robot control strategies like probabilistic filters such as Kalman Filter or Particle Filter can be used to estimate the control input from the obtained rotational velocity so to overcome uncertainties in the navigation due to the poor performance of feature matcher/-tracker, or the ill-conditioning cases occurred due to few textures in the scene.

With the incorporation of the above mentioned perspectives, we could have a generalized framework for the image-based indoor navigation of Pioneer like robots.

References

Bibliography

[AC09] G. Alenya and J. L. Crowley, "Time to contact for obstacle avoidance," in *Proceedings. European Conference on Mobile Robotics, (ECMR 2009)*. KoREMA, 2009, pp. 19–24. 3.2.3

[AFDM08] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer, "Fast and incremental method for loop-closure detection using bags of visual words," *IEEE Transactions on Robotics, (T-RO)*, vol. 24, no. 5, pp. 1027–1037, 2008. 3.4.3

[Agi79] G. J. Agin, *Real time control of a robot with a mobile camera*. SRI International, 1979. 2.3

[AOV12] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in *Proceedings. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR 2012)*. IEEE, 2012, pp. 510–517. 2.1.2.3

[AT11] C. Akinlar and C. Topal, "Edlines: A real-time line segment detector with a false detection control," *Pattern Recognition Letters*, vol. 32, no. 13, pp. 1633–1642, 2011. 2.1.2.2, 6.1, 6.2.1, 7.2

[BD14] C. Beall and F. Dellaert, "Appearance-based localization across seasons in a metric map," in *Proceedings. Workshop on Planning, Perception and Navigation for Intelligent Vehicles, (PPNI 2014)*, 2014. 3.2.1

[BDW06] T. Bailey and H. Durrant-Whyte, "Simultaneous Localization And Mapping (SLAM): Part II," *IEEE Robotics & Automation Magazine, (RAM)*, vol. 13, no. 3, pp. 108–117, 2006. 2.1.5

[BEF96] J. Borenstein, H. R. Everett, and L. Feng, *Navigating Mobile Robots: Systems and Techniques*. A. K. Peters, Ltd., 1996. 3.2.1

[BETVG08] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (surf),” *Computer Vision and Image Understanding, (CVIU)*, vol. 110, no. 3, pp. 346–359, 2008. 2.1.2.2, 2.7, 2.1.2.3, 7.2

[BFOO08] F. Bonin-Font, A. Ortiz, and G. Oliver, “Visual navigation for mobile robots: A survey,” *Journal of Intelligent and Robotic Systems*, vol. 53, no. 3, pp. 263–296, 2008. 3, 3.1.1, 3.2

[BFVG05] H. Bay, V. Ferraris, and L. Van Gool, “Wide-baseline stereo matching with line segments,” in *Proceedings. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR 2005)*, vol. 1. IEEE, 2005, pp. 329–336. 2.1.2.3

[BGL] J. R. Beveridge, C. Graves, and C. Lesher, “Some lessons learned from coding the burns line extraction algorithm in the DARPA image understanding environment,” in Technical Report CS-96-125, Computer Science Departement, Colorado State University, USA, 1996. 2.1.2.2

[BHR86] J. B. Burns, A. R. Hanson, and E. M. Riseman, “Extracting straight lines,” *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, no. 4, pp. 425–455, 1986. 2.1.2.2

[Bou] J.-Y. Bouguet, “Camera calibration toolbox for Matlab,” accessed: 2016-09-08. [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/ 2.1.1.2

[BRGC16a] S. R. Bista, P. Robuffo Giordano, and F. Chaumette, “Appearance-based indoor navigation by IBVS using line segments,” *IEEE Robotics and Automation Letters, (RA-L)*, vol. 1, no. 1, pp. 423–430, January 2016, also presented in IEEE International Conference on Robotics and Automation, (ICRA 2016). 1.4, 6

[BRGC16b] ——, “Appearance-based indoor navigation by IBVS using mutual information,” in *Proceedings. IEEE International Conference on Control, Automation, Robotics and Vision, (ICARCV 2016)*, November 2016. 1.4, 5

[BRGCew] ——, “Combining line segments and points for appearance-based indoor navigation by image based visual servoing,” *Submitted to IEEE International Conference on Robotics and Automation, (ICRA 2017)*, Under Review. 1.4, 7

[BSBB06] M. Bennewitz, C. Stachniss, W. Burgard, and S. Behnke, “Metric localization with scale-invariant visual features using a single perspective camera,” in *Proceedings. European Robotics Symposium*. Springer, 2006, pp. 195–209. 3.2.1

[BSC09] B. Bacca, J. Salvi, and X. Cufí, “Appearance-based slam for mobile robots,” in *Proceedings. International Conference of the Catalan Association for Artificial Intelligence*, 2009, pp. 55–64. 3.5.3

[BSMH14] H. M. Becerra, C. Sagüés, Y. Mezouar, and J.-B. Hayet, “Visual navigation of wheeled mobile robots using direct feedback of a geometric constraint,” *Autonomous Robots*, vol. 37, no. 2, pp. 137–156, 2014. 3.5.6

[BTZK07] O. Booij, B. Terwijn, Z. Zivkovic, and B. Krose, “Navigation using an appearance based topological map,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2007)*. IEEE, 2007, pp. 3927–3932. 3.2.1, 3.3.2, 3.4.2, 3.5.3, 3.5.6

[CB09] Z. Chen and S. T. Birchfield, “Qualitative vision-based path following,” *IEEE Transactions on Robotics, (T-RO)*, vol. 25, no. 3, pp. 749–754, 2009. 3.5.6

[CC09] A. Cherubini and F. Chaumette, “Visual navigation with a time-independent varying reference,” in *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2009)*. IEEE, 2009, pp. 5968–5973. 3.5.6

[CC13] ——, “Visual navigation of a mobile robot with laser-based collision avoidance,” *The International Journal of Robotics Research, (IJRR)*, vol. 32, no. 2, pp. 189–205, 2013. (document), 4, 8.2, 8.2

[CCO08] A. Cherubini, F. Chaumette, and G. Oriolo, “An image-based visual servoing scheme for following paths with nonholonomic mobile robots,” in *Proceedings. IEEE International Conference on Control, Automation, Robotics and Vision, (ICARCV 2008)*. IEEE, 2008, pp. 108–113. 4

[CH06] F. Chaumette and S. Hutchinson, “Visual servo control. i. basic approaches,” *IEEE Robotics & Automation Magazine, (RAM)*, vol. 13, no. 4, pp. 82–90, 2006. (document), 1.2, 2.3, 2.3, 2.3.1, 2.3.2, 4, 4, 4, 5.2.3.2, 6.2.3.2, 7.2.2.2

[CH07] ——, “Visual servo control, part ii: Advanced approaches,” *IEEE Robotics & Automation Magazine, (RAM)*, vol. 14, no. 1, pp. 109–118, 2007. 2.3, 2.3.2, 2.3.3, 6.2.3.2

[CK98] N. Chiba and T. Kanade, “A tracker for broken and closely-spaced lines,” *International Archives of Photogrammetry and Remote Sensing*, vol. 32, pp. 676–683, 1998. 2.1.2.4

[CLO$^+$12] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua, “BRIEF: Computing a local binary descriptor very fast,” *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 34, no. 7, pp. 1281–1298, 2012. 2.1.2.3

[CM11] C. Collewet and E. Marchand, “Photometric visual servoing,” *IEEE Transactions on Robotics, (T-RO)*, vol. 27, no. 4, pp. 828–834, 2011. 5.1

[CMC08] C. Collewet, E. Marchand, and F. Chaumette, “Visual servoing set free from image processing,” in *IEEE International Conference on Robotics and Automation, (ICRA 2008)*, 2008, pp. 81–86. 5.1

[CMEM09] J. Courbon, Y. Mezouar, L. Eck, and P. Martinet, “A generic framework for topological navigation of urban vehicle,” in *ICRA Workshop on Safe navigation in open and dynamic environments Application to autonomous vehicles*. IEEE, 2009. 3.1.2

[CMM08] J. Courbon, Y. Mezouar, and P. Martinet, “Indoor navigation of a nonholonomic mobile robot using a visual memory,” *Autonomous Robots, (AR)*, vol. 25, no. 3, pp. 253–266, 2008. 3.2.1, 3.3.2, 3.4.2, 3.5.6

[CMM09] ——, “Autonomous navigation of vehicles from a visual memory using a generic camera model,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 3, pp. 392–402, 2009. 3.1.2, 3.5.5, 3.5.6, 5.1

[CN08] M. Cummins and P. Newman, “Fab-map: Probabilistic localization and mapping in the space of appearance,” *The International Journal of Robotics Research, (IJRR)*, vol. 27, no. 6, pp. 647–665, 2008. 3.5.4

[CN11] ——, “Appearance-only slam at large scale with fab-map 2.0,” *The International Journal of Robotics Research, (IJRR)*, vol. 30, no. 9, pp. 1100–1123, 2011. 2.1.5, 3.2.2, 3.3.2, 3.5.4

[Cor93] P. I. Corke, “Visual control of robot manipulators-a review,” *Visual Servoing*, vol. 7, pp. 1–31, 1993. 2.3

[Cor11] P. Corke, *Robotics, Vision and Control: fundamental algorithms in MATLAB*. Springer, 2011, vol. 73. 2.3.2, 2.3.2

[CV99] D. Chetverikov and J. Verestói, “Feature point tracking for incomplete trajectories,” *Computing*, vol. 62, no. 4, pp. 321–338, 1999. 2.1.2.4

[CY96] T. Chan and R. K. Yip, "Line detection algorithm," in *Proceedings. International Conference on Pattern Recognition, (ICPR 1996)*, vol. 2. IEEE, 1996, pp. 126–130. 2.1.2.2

[Dam10] A. Dame, "A unified direct approach for visual servoing and visual tracking using mutual information," Ph.D. dissertation, Université de Rennes 1, 2010. 2.3.2

[DF90] R. Deriche and O. Faugeras, "Tracking line segments," in *Proceedings. European Conference on Computer Vision, (ECCV 1990)*. Springer, 1990, pp. 259–268. 2.1.2.4

[DH72] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972. 2.1.2.2

[DK02] G. N. Desouza and A. C. Kak, "Vision for mobile robot navigation: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 24, no. 2, pp. 237–267, 2002. 3, 3.2

[DM12] A. Dame and E. Marchand, "Second-order optimization of mutual information for real-time image registration," *IEEE Transactions on Image Processing, (T-IP)*, vol. 21, no. 9, pp. 4190–4203, 2012. 5.1, 5.2.1

[DM13] ——, "Using mutual information for appearance-based visual path following," *Robotics and Autonomous Systems, (RAS)*, vol. 61, no. 3, pp. 259–270, 2013. 1.4, 3.1.2, 3.3.2, 3.5.2, 5, 5.1, 5.1, 5.2.1, 5.2.1, 5.2.2, 5.2.3.2, 5.2.3.2

[DMM07] A. Desolneux, L. Moisan, and J.-M. Morel, *From Gestalt Theory to Image Analysis: A Probabilistic Approach*, 1st ed. Springer Publishing Company, Incorporated, 2007. 2.1.2.2

[DRMS07] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 29, no. 6, pp. 1052–1067, 2007. 2.1.2.4, 3.2.2, 3.3.1, 6.1

[DSRC11] A. Diosi, S. Segvic, A. Remazeilles, and F. Chaumette, "Experimental evaluation of autonomous driving based on visual memory and image-based visual servoing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 3, pp. 870–883, 2011. (document), 1.2, 2.1.2.4, 3.1.2, 3.3.2, 3.5.3, 3.5.6, 6.1, 6.3.4, 6.17, 7.2, 7.2.1, 7.2.2.2, 7.3.2.5, 8.1

[DWB06] H. Durrant-Whyte and T. Bailey, “Simultaneous Localization And Mapping: Part I,” *IEEE Robotics & Automation Magazine, (RAM)*, vol. 13, no. 2, pp. 99–110, June 2006. 2.1.5

[DYO05] N. X. Dao, B.-J. You, and S.-R. Oh, “Visual navigation for indoor mobile robots using a single camera,” in *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2005)*, 2005, pp. 1992–1997. 6.1

[ECR92] B. Espiau, F. Chaumette, and P. Rives, “A new approach to visual servoing in robotics,” *IEEE Transactions on Robotics and Automation*, vol. 8, no. 3, pp. 313–326, June 1992. 2.3.2, 6.2.3.2, 6.2.3.2

[Elf89] A. Elfes, “Occupancy grids: A probabilistic framework for robot perception and navigation,” Ph.D. dissertation, Pittsburgh, PA, USA, 1989. 3.4.1

[ESC14] J. Engel, T. Schöps, and D. Cremers, “LSD-SLAM: Large-scale direct monocular SLAM,” in *Proceedings. European Conference on Computer Vision, (ECCV 2014)*. Springer, 2014, pp. 834–849. 2.1.5, 3.2.2

[FB81] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981. 2.1.3.4

[FEN07] F. Fraundorfer, C. Engels, and D. Nistér, “Topological mapping, localization and navigation using image collections,” in *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2007)*. IEEE, 2007, pp. 3872–3877. 3.5.4

[FM03] D. Filliat and J.-A. Meyer, “Map-based navigation in mobile robots:: I. a review of localization strategies,” *Cognitive Systems Research*, vol. 4, no. 4, pp. 243–282, 2003. 3.4.3

[FO08] L. A. Fernandes and M. M. Oliveira, “Real-time line detection through an improved hough transform voting scheme,” *Pattern Recognition, (PR)*, vol. 41, no. 1, pp. 299–314, 2008. 2.1.2.2

[FOPV13] A. Faragasso, G. Oriolo, A. Paolillo, and M. Vendittelli, “Vision-based corridor navigation for humanoid robots,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2013)*. IEEE, 2013, pp. 3190–3195. 2.2.1, 3.2.3, 6.1

[FPRARM15] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual Simultaneous Localization And Mapping: A Survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 55–81, 2015. 2.1.5, 3, 3.3.1

[FS12] F. Fraundorfer and D. Scaramuzza, "Visual Odometry: Part II: Matching, robustness, optimization, and applications," *IEEE Robotics & Automation Magazine, (RAM)*, vol. 19, no. 2, pp. 78–90, 2012. 2.1.6

[FWH10] B. Fan, F. Wu, and Z. Hu, "Line matching leveraged by point correspondences," in *Proceedings. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR 2010)*. IEEE, 2010, pp. 390–397. 2.1.2.4, 6.1

[GFF08] J.-S. Gutmann, M. Fukuchi, and M. Fujita, "3d perception and environment map generation for humanoid robot navigation," *The International Journal of Robotics Research, (IJRR)*, vol. 27, no. 10, pp. 1117–1134, 2008. 3.4.1

[GFO15] E. Garcia-Fidalgo and A. Ortiz, "Vision-based topological mapping and localization methods: A survey," *Robotics and Autonomous Systems, (RAS)*, vol. 64, pp. 1–20, 2015. 3, 3.1.1, 3.4.2

[GJMR10] v. G. R. Grompone, J. Jakubowicz, J.-M. Morel, and G. Randall, "Lsd: a fast line segment detector with a false detection control." *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 32, no. 4, pp. 722–732, 2010. 2.1.2.2, 6.1

[GKSB10] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010. 2.1.5

[GMC06] A. Gee and W. Mayol-Cuevas, "Real-time model-based SLAM using line segments," *Advances in Visual Computing*, pp. 354–363, 2006. 3.2.2, 3.3.1, 6.1

[GMMW10] A. J. Glover, W. P. Maddern, M. J. Milford, and G. F. Wyeth, "Fab-map+ ratslam: Appearance-based slam for multiple times of day," in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2010)*. IEEE, 2010, pp. 3507–3512. 2.1.5, 3.1.2, 3.2.2, 3.3.2, 3.5.2

[GNTVG07] T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool, "Omnidirectional vision based topological navigation," *International Journal of Computer Vision, (IJCV)*, vol. 74, no. 3, pp. 219–236, 2007. 3.3.2, 3.5.5, 3.5.6

[GWSV00] J. Gaspar, N. Winters, and J. Santos-Victor, "Vision-based navigation and environmental representations with an omnidirectional camera," *IEEE Transactions on Robotics and Automation, (T-RO)*, vol. 16, no. 6, pp. 890–898, 2000. 3.5.2

[HAA16] M. Hassaballah, A. A. Abdelmgeid, and H. A. Alshazly, "Image features detection, description and matching," in *Image Feature Detectors and Descriptors*. Springer, 2016, pp. 11–45. 2.1.2.1

[Har97] R. I. Hartley, "In defense of the eight-point algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 19, no. 6, pp. 580–593, 1997. 2.1.3.4, 2.1.4.1

[HHC96] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, 1996. 2.3

[HS81] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence, (AI)*, vol. 17, no. 1-3, pp. 185–203, 1981. 2.1.2.4

[HS88] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey Vision Conference, (AVC 1988)*, vol. 15. Citeseer, 1988, p. 50. 2.1.2.2, 2.6

[HS12] K. Hirose and H. Saito, "Fast line description for line-based slam," in *Proceedings. British Machine Vision Conference, (BMVC 2012)*, 2012. 3.2.2

[HZ03] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge university press, 2003. 2.1.1.1, 2.1.1.2, 2.1.1.2, 2.1.1.2, 2.1.2.4, 2.9, 2.10, 2.1.3.2, 2.11, 2.1.3.3, 2.1.3.3, 2.1.3.4, 2.1.3.4, 2.1.4.1, 2.1.4.1, 2.1.4.2, 3.3.1, 6.2.1

[JA77] G. J. Agin, "Servoing with visual feedback," in *Proceedings. International Symposium On Industrial Robots, (ISIR 1977)*, 1977, pp. 551–560. 2.3

[JY13] E. Johns and G.-Z. Yang, "Feature co-occurrence maps: Appearance-based localisation throughout the day," in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2013)*. IEEE, 2013, pp. 3212–3218. 3.5.4

[KDCA07] W.-j. KANG, X.-m. DING, J.-w. CUI, and L. AO, "Fast straight-line extraction algorithm based on improved hough transform," *Opto-Electronic Engineering*, vol. 3, p. 023, 2007. 2.1.2.2

[KEB91] N. Kiryati, Y. Eldar, and A. M. Bruckstein, "A probabilistic hough transform," *Pattern recognition, (PR)*, vol. 24, no. 4, pp. 303–316, 1991. 2.1.2.2

[KG96] H. K. Khalil and J. Grizzle, *Nonlinear Systems*. Prentice hall New Jersey, 1996, vol. 3. 2.2.2.2

[KL04] J. Kosecká and F. Li, “Vision based topological markov localization,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2004)*, vol. 2. IEEE, 2004, pp. 1481–1486. 3.5.3

[KM07] G. Klein and D. Murray, “Parallel tracking and mapping for small AR workspaces,” in *IEEE and ACM International Symposium on Mixed and Augmented Reality, (ISMAR 2007)*. IEEE, 2007, pp. 225–234. 3.4.1

[KOA14] Y. Kuang, M. Oskarsson, and K. Astrom, “Revisiting trifocal tensor estimation using lines,” in *Proceedings. International Conference on Pattern Recognition, (ICPR 2014)*. IEEE Computer Society, 2014, pp. 2419–2423. 2.1.3.3

[KTTH11] A. Kawewong, N. Tongprasit, S. Tangruamsub, and O. Hasegawa, “Online and incremental appearance-based slam in highly dynamic environments,” *The International Journal of Robotics Research, (IJRR)*, vol. 30, no. 1, pp. 33–55, 2011. 3.5.3

[KZBD03] J. Kosecka, L. Zhou, P. Barber, and Z. Duric, “Qualitative image based localization in indoors environments,” in *Proceedings. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR 2003)*, vol. 2. IEEE, 2003, pp. 3–8. 3.3.2, 3.5.2

[Lab07] F. Labrosse, “Short and long-range visual navigation using warped panoramic images,” *Robotics and Autonomous Systems, (RAS)*, vol. 55, no. 9, pp. 675–684, 2007. 3.3.2

[LCS11] S. Leutenegger, M. Chli, and R. Y. Siegwart, “BRISK: Binary robust invariant scalable keypoints,” in *Proceedings. International Conference on Computer Vision, (ICCV 2011)*. IEEE Computer Society, 2011, pp. 2548–2555. 2.1.2.2, 2.1.2.3

[LDW91] J. J. Leonard and H. F. Durrant-Whyte, “Mobile robot localization by tracking geometric beacons,” *IEEE Transactions on Robotics and Automation, (T-RO)*, vol. 7, no. 3, pp. 376–382, 1991. 2.2.1

[Lin98] T. Lindeberg, “Feature detection with automatic scale selection,” *International Journal of Computer Vision, (IJCV)*, vol. 30, no. 2, pp. 79–116, 1998. 2.1.2.2

[LK^{+}81] B. D. Lucas, T. Kanade *et al.*, “An iterative image registration technique with an application to stereo vision,” in *Proceedings. International Joint Conference on Artificial Intelligence, (IJCAI 1981)*, vol. 81, no. 1, 1981, pp. 674–679. 2.1.2.4

[LK06] F. Li and J. Kosecka, “Probabilistic location recognition using reduced feature set,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2006)*. IEEE, 2006, pp. 3405–3410. 3.5.3

[LLL94] L. T. Lam, W. C. Lam, and D. N. Leung, “A knowledge-based boundary convergence algorithm for line detection,” *Pattern recognition letters*, vol. 15, no. 4, pp. 383–392, 1994. 2.1.2.2

[LM04] J. Lapreste and Y. Mezouar, “A hessian approach to visual servoing,” in *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2004)*, vol. 1, 2004, pp. 998–1003. 5.2.3.2

[Low04] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision, (IJCV)*, vol. 60, no. 2, pp. 91–110, 2004. 2.1.2.2, 2.1.2.3, 2.7, 2.1.2.4

[LP02] B. Liang and N. Pears, “Visual navigation using planar homographies,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2002)*, vol. 1. IEEE, 2002, pp. 205–210. 3.2.3

[LZLS13] J. H. Lee, G. Zhang, J. Lim, and I. H. Suh, “Place recognition using straight lines for vision-based slam,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2013)*. IEEE, 2013, pp. 3799–3806. 3.5.4

[MAMT15] R. Mur-Artal, J. Montiel, and J. D. Tardós, “ORB-SLAM: a versatile and accurate monocular SLAM system,” *IEEE Transactions on Robotics, (T-RO)*, vol. 31, no. 5, pp. 1147–1163, 2015. 3.2.2

[MCB99] E. Malis, F. Chaumette, and S. Boudet, “21/2d visual servoing,” *IEEE Transactions on Robotics and Automation, (T-RO)*, vol. 15, no. 2, pp. 238–250, 1999. 2.3.3

[MCUP04] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide-baseline stereo from maximally stable extremal regions,” *Image and Vision Computing*, vol. 22, no. 10, pp. 761–767, 2004. 2.1.2.2

[MCV$^+$97] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens, "Multimodality image registration by maximization of mutual information," *IEEE Transactions on Medical Imaging*, vol. 16, no. 2, pp. 187–198, 1997. 5.2.1

[MGH$^+$95] L. Matthies, E. Gat, R. Harrison, B. Wilcox, R. Volpe, and T. Litwin, "Mars microrover navigation: Performance evaluation and enhancement," *Autonomous robots, (AR)*, vol. 2, no. 4, pp. 291–311, 1995. 3.1.2

[MGK00] J. Matas, C. Galambos, and J. Kittler, "Robust detection of lines using the progressive probabilistic hough transform," *Computer Vision and Image Understanding, (CVIU)*, vol. 78, no. 1, pp. 119–137, 2000. 2.1.2.2

[MGS07] A. C. Murillo, J. J. Guerrero, and C. Sagues, "SURF features for efficient robot localization with omnidirectional images," in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2007)*. IEEE, 2007, pp. 3901–3907. 3.5.5

[MHV$^+$01] D. Mattes, D. R. Haynor, H. Vesselle, T. K. Lewellyn, and W. Eubank, "Nonrigid multimodality image registration," in *Proceedings. SPIE: Medical Imaging*, 2001, pp. 1609–1620. 5.1

[MII96a] Y. Matsumoto, M. Inaba, and H. Inoue, "Visual navigation using view-sequenced route representation," in *IEEE International Conference on Robotics and Automation, (ICRA 1996)*, vol. 1, 1996, pp. 83–88. 5.1

[MII96b] ——, "Visual navigation using view-sequenced route representation," in *Proceedings. IEEE International Conference onRobotics and Automation, (ICRA 1996)*, vol. 1. IEEE, 1996, pp. 83–88. 3.5.6

[MIII99] Y. Matsumoto, K. Ikeda, M. Inaba, and H. Inoue, "Visual navigation using omnidirectional view sequence," in *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 1999)*, vol. 1. IEEE, 1999, pp. 317–322. 3.5.6

[MK04] G. Medioni and S. B. Kang, *Emerging Topics in Computer Vision*. Prentice Hall PTR, 2004. 2.1.1.2, 2.1.1.2

[ML04] T. Mitchell and F. Labrosse, "Visual homing: a purely appearance-based approach," in *Proceedings. Towards Autonomous Robotic Systems*, 2004, pp. 101–108. 3.5.2

[ML12] M. Muja and D. G. Lowe, "Fast matching of binary features," in *Proceedings. Conference on Computer and Robot Vision, (CRV 2012)*. IEEE, 2012, pp. 404–410. 2.1.2.4

[ML14] ——, “Scalable nearest neighbor algorithms for high dimensional data,” *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 36, 2014. 2.1.2.4

[MMW12] W. Maddern, M. Milford, and G. Wyeth, “Towards persistent indoor appearance-based localization, mapping and navigation using cat-graph,” in *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2012)*. IEEE, 2012, pp. 4224–4230. 2.1.5, 3.2.2, 3.3.2

[Mor77] H. P. Morevec, “Towards automatic visual obstacle avoidance,” in *Proceedings. International Joint Conference on Artificial Intelligence, (IJCAI 1977)*, vol. 2. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1977, pp. 584–584. 2.1.2.2

[Mor90] H. P. Moravec, “The stanford cart and the cmu rover,” in *Autonomous Robot Vehicles*. Springer, 1990, pp. 407–419. 3.2.2

[MS04] K. Mikolajczyk and C. Schmid, “Scale & affine invariant interest point detectors,” *International Journal of Computer Vision, (IJCV)*, vol. 60, no. 1, pp. 63–86, 2004. 2.1.2.2

[MSC05] É. Marchand, F. Spindler, and F. Chaumette, “Visp for visual servoing: a generic software platform with a wide class of robot control skills,” *IEEE Robotics & Automation Magazine, (RAM)*, vol. 12, no. 4, pp. 40–52, 2005. 2.1.1.2, 4

[MSG$^+$07] A. Murillo, C. Sagüés, J. J. Guerrero, T. Goedemé, T. Tuytelaars, and L. Van Gool, “From omnidirectional images to hierarchical localization,” *Robotics and Autonomous Systems, (RAS)*, vol. 55, no. 5, pp. 372–382, 2007. 3.5.5

[MTS$^+$05] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool, “A comparison of affine region detectors,” *International Journal of Computer Vision, (IJCV)*, vol. 65, no. 1-2, pp. 43–72, 2005. 2.1.2.2, 2.1.2.2, 2.1.2.3

[MW10] M. Milford and G. Wyeth, “Persistent navigation and mapping using a biologically inspired slam system,” *The International Journal of Robotics Research, (IJRR)*, vol. 29, no. 9, pp. 1131–1153, 2010. 2.1.5, 3.2.2, 3.3.2

[MW15] B. Micusik and H. Wildenauer, “Descriptor free visual indoor localization with line segments,” in *Proceedings. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR 2015)*. IEEE, 2015. 6.1, 7.1

[MWP04] M. J. Milford, G. F. Wyeth, and D. Prasser, "Ratslam: a hippocampal model for simultaneous localization and mapping," in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2004)*, vol. 1. IEEE, 2004, pp. 403–408. 3.5.2

[MZPI04] E. Menegatti, M. Zoccarato, E. Pagello, and H. Ishiguro, "Image-based monte carlo localisation with omnidirectional images," *Robotics and Autonomous Systems, (RAS)*, vol. 48, no. 1, pp. 17–30, 2004. 3.5.2

[Nis04] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 26, no. 6, pp. 756–770, 2004. 2.1.4.1, 7.2

[NPVCL08] P. Neubert, P. Protzel, T. Vidal-Calleja, and S. Lacroix, "A fast visual line segment tracker," in *Proceedings. IEEE International Conference on Emerging Technologies and Factory Automation*. IEEE, 2008, pp. 353–360. 2.1.2.4

[NSP13] P. Neubert, N. Sunderhauf, and P. Protzel, "Appearance change prediction for long-term navigation across seasons," in *Proceedings. European Conference on Mobile Robots, (ECMR 2013)*. IEEE, 2013, pp. 198–203. 3.1.2

[Ope] OpenCV, "Camera calibration with opencv," accessed: 2016-09-08. [Online]. Available: http://docs.opencv.org/2.4/doc/tutorials/calib3d/camera_calibration/camera_calibration.html 2.1.1.2

[PKB14] F. Pasteau, A. Krupa, and M. Babel, "Vision-based assistance for wheelchair navigation along corridors," in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2014)*. IEEE, 2014, pp. 4430–4435. 3.2.3, 6.1

[PKVVG00] M. Pollefeys, R. Koch, M. Vergauwen, and L. Van Gool, "Automated reconstruction of 3d scenes from sequences of images," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 55, no. 4, pp. 251–267, 2000. 2.1.4.2

[PMV98] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever, "A multiscale approach to mutual information matching," in *Proceedings. SPIE: Medical Imaging*, 1998, pp. 1334–1344. 5.2.1

[PMV03] J. P. Pluim, J. A. Maintz, and M. A. Viergever, "Mutual-information-based registration of medical images: a survey," *IEEE Transactions on Medical Imaging*, vol. 22, no. 8, pp. 986–1004, 2003. 5.1

[PN10] R. Paul and P. Newman, “Fab-map 3d: Topological mapping with spatial and visual appearance,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2010)*. IEEE, 2010, pp. 2649–2656. 3.5.4

[Pol99] M. Pollefeys, “Self-calibration and metric 3d reconstruction from uncalibrated image sequences,” Ph.D. dissertation, ESAT-PSI, KU Leuven, 1999. 2.1.4.2

[RC07] A. Remazeilles and F. Chaumette, “Image-based robot navigation from an image memory,” *Robotics and Autonomous Systems, (RAS)*, vol. 55, no. 4, pp. 345–356, 2007. 3.2.1, 3.4.2, 3.4.3

[RLDL07] E. Royer, M. Lhuillier, M. Dhome, and J.-M. Lavest, “Monocular vision for mobile robot localization and autonomous navigation,” *International Journal of Computer Vision, (IJCV)*, vol. 74, no. 3, pp. 237–260, 2007. 3.2.1, 3.3.1, 3.5.6

[RMC06] A. Remazeilles, N. Mansard, and F. Chaumette, “A qualitative visual servoing to ensure the visibility constraint,” in *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2006)*. IEEE, 2006, pp. 4297–4303. 3.5.6

[RMG14] A. Rituerto, A. Murillo, and J. Guerrero, “Semantic labeling for indoor topological mapping using a wearable catadioptric system,” *Robotics and Autonomous Systems, (RAS)*, vol. 62, no. 5, pp. 685–695, 2014. 3.5.2

[RPD10] E. Rosten, R. Porter, and T. Drummond, “Faster and better: A machine learning approach to corner detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 32, no. 1, pp. 105–119, 2010. 2.1.2.2, 2.6, 2.1.2.2, 7.2

[RRKB11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” in *Proceedings. International Conference on Computer Vision, (ICCV 2011)*. IEEE Computer Society, 2011, pp. 2564–2571. 2.1.2.2, 2.1.2.3

[RTA$^+$09] A. Ramisa, A. Tapus, D. Aldavert, R. Toledo, and R. L. De Mantaras, “Robust vision-based robot localization using combinations of local feature region detectors,” *Autonomous Robots, (AR)*, vol. 27, no. 4, pp. 373–385, 2009. 3.5.3

[RZL03] P. E. Rybski, F. Zacharias, and J.-F. Lett, “Using visual features to build topological maps of indoor environments,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2003)*, vol. 1. IEEE, 2003, pp. 850–855. 3.5.3

[SB97] S. M. Smith and J. M. Brady, “SUSAN-a new approach to low level image processing,” *International Journal of Computer Vision, (IJCV)*, vol. 23, no. 1, pp. 45–78, 1997. 2.1.2.2, 2.1.2.2

[SEN06] H. Stewenius, C. Engels, and D. Nistér, “Recent developments on direct relative orientation,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 60, no. 4, pp. 284–294, 2006. 3

[SF11] D. Scaramuzza and F. Fraundorfer, “Visual Odometry [tutorial],” *IEEE Robotics & Automation Magazine, (RAM)*, vol. 18, no. 4, pp. 80–92, 2011. 2.1.6

[SHV06] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot modeling and control*. Wiley New York, 2006, vol. 3. 2.2.2.3, 2.3.2

[SI73] Y. Shirai and H. Inoue, “Guiding a robot by visual feedback in assembling tasks,” *Pattern Recognition, (PR)*, vol. 5, no. 2, pp. 99–108, 1973. 2.3

[SLL02] S. Se, D. Lowe, and J. Little, “Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks,” *The International Journal of robotics Research, (IJRR)*, vol. 21, no. 8, pp. 735–758, 2002. 3.4.1

[SNS11] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*. The MIT Press, 2011. 2.13, 2.14, 2.2.2.2, 2.2.2.2, 2.15

[SRD06] P. Smith, I. Reid, and A. Davison, “Real-time monocular SLAM with straight lines,” in *Proceedings. British Machine Vision Conference, (BMVC 2006)*, 2006, pp. 17–26. 3.2.2, 6.1

[ŠRDC09] S. Šegvić, A. Remazeilles, A. Diosi, and F. Chaumette, “A mapping and localization framework for scalable appearance-based navigation,” *Computer Vision and Image Understanding, (CVIU)*, vol. 113, no. 2, pp. 172–187, 2009. (document), 1.2, 3.2.1, 3.3.2, 3.5.3, 3.5.6, 4, 6, 6.1, 6.3.4, 6.17, 8.1

[SSVO10] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer Science & Business Media, 2010. 2.2.2.1

[SVSCG93] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi, “Divergent stereo for robot navigation: Learning from bees,” in *Proceedings. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR 1993)*. IEEE, 1993, pp. 434–439. 3.2.3

[SZ97] C. Schmid and A. Zisserman, “Automatic line matching across views,” in *Proceedings. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR 1997)*. IEEE, 1997, pp. 666–671. 2.1.2.4

[SZ00] ——, “The geometry and matching of lines and curves over multiple views,” *International Journal of Computer Vision, (IJCV)*, vol. 40, no. 3, pp. 199–233, 2000. 6.1

[Sze10] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010. 2.1.1.1, 2.1.1.2, 2.1.1.2, 2.1.1.2, 2.1.2, 2.1.2.1, 2.1.4.1

[TA12] C. Topal and C. Akinlar, “Edge Drawing: A combined real-time edge and segment detector,” *Journal of Visual Communication and Image Representation*, vol. 23, no. 6, pp. 862–872, 2012. 2.1.2.2

[Thr98] S. Thrun, “Learning metric-topological maps for indoor mobile robot navigation,” *Artificial Intelligence, (AI)*, vol. 99, no. 1, pp. 21–71, 1998. 3.4.1

[TMFR03] A. Torralba, K. Murphy, W. Freeman, and M. Rubin, “Context-based vision system for place and object recognition,” in *Proceedings. IEEE International Conference on Computer Vision, (ICCV 2003)*, vol. 1, 2003, pp. 273–280. 2.1.2.1

[TMHF99] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, “Bundle adjustment-a modern synthesis,” in *International Workshop on Vision Algorithms*. Springer, 1999, pp. 298–372. 2.1.4, 2.1.4.2, 2.1.5

[TNS01] N. Tomatis, I. Nourbakhsh, and R. Siegwart, “Simultaneous localization and map building: A global topological model with local metric maps,” in *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2001)*, vol. 1. IEEE, 2001, pp. 421–426. 3.4.3

[UN00] I. Ulrich and I. Nourbakhsh, “Appearance-based place recognition for topological localization,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2000)*, vol. 2. IEEE, 2000, pp. 1023–1029. 3.5.2

[VLF04] L. Vacchetti, V. Lepetit, and P. Fua, “Stable real-time 3d tracking using online and offline information,” *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 26, no. 10, pp. 1385–1391, 2004. 3.3.1

[VSSV00] R. F. Vassallo, H. J. Schneebeli, and J. Santos-Victor, “Visual servoing and appearance for navigation,” *Robotics and Autonomous System, (RAS)*, vol. 31, no. 1, pp. 87–97, 2000. 3.2.1, 3.2.3, 3.5.6, 5.1, 6.1

[VTVG14] B. Verhagen, R. Timofte, and L. Van Gool, “Scale-invariant line descriptors for wide baseline matching,” in *Proceedings. IEEE Winter Conference on Applications of Computer Vision*. IEEE, 2014, pp. 493–500. 2.1.2.3

[VWI97] P. Viola and W. M. Wells III, “Alignment by maximization of mutual information,” *International Journal of Computer Vision, (IJCV)*, vol. 24, no. 2, pp. 137–154, 1997. 5.1, 5.2.1

[WCH^{+}92] J. Weng, P. Cohen, M. Herniou *et al.*, “Camera calibration with distortion models and accuracy evaluation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 14, no. 10, pp. 965–980, 1992. 2.1.1.2

[WCZ05] J. Wang, R. Cipolla, and H. Zha, “Vision-based global localization using a visual vocabulary,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2005)*. IEEE, 2005, pp. 4230–4235. 3.5.4

[WNY09] L. Wang, U. Neumann, and S. You, “Wide-baseline image matching using line signatures,” in *Proceedings. IEEE International Conference on Computer Vision, (ICCV 2009)*. IEEE, 2009, pp. 1311–1318. 2.1.2.3

[WSM08] F. Werner, J. Sitte, and F. Maire, “Visual topological mapping and localisation using colour histograms,” in *Proceedings. International Conference on Control, Automation, Robotics and Vision, (ICARCV 2008)*. IEEE, 2008, pp. 341–346. 3.3.2, 3.5.2

[WTMZ07] C. Weiss, H. Tamimi, A. Masselli, and A. Zell, “A hybrid approach for vision-based outdoor robot localization using global and local image features,” in *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2007)*. IEEE, 2007, pp. 1047–1052. 3.5.5

[WWH09] Z. Wang, F. Wu, and Z. Hu, “MSLD: A robust descriptor for line matching,” *Pattern Recognition, (PR)*, vol. 42, no. 5, pp. 941–953, 2009. 2.1.2.3, 2.8, 2.1.2.3

[WY12] J. Wang and Y. Yagi, “Robust location recognition based on efficient feature integration,” in *Proceedings. IEEE International Conference on Robotics and Biomimetics, (ROBIO 2012)*. IEEE, 2012, pp. 97–101. 3.5.5

[WY13] ——, “Efficient topological localization using global and local feature matching,” *International Journal of Advanced Robotic Systems*, vol. 10, 2013. 3.5.5

[WZC06] J. Wang, H. Zha, and R. Cipolla, “Coarse-to-fine vision-based localization by indexing scale-invariant features,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 36, no. 2, pp. 413–422, 2006. 3.5.4

[WZG14] J. Wu, H. Zhang, and Y. Guan, “An efficient visual loop closure detection method in a map of 20 million key locations,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2014)*. IEEE, 2014, pp. 861–866. 3.5.2

[XBH14] D. Xu, H. Badino, and D. Huber, “Topometric localization on a road network,” in *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2014)*. IEEE, 2014, pp. 3448–3455. 3.5.2

[XOK09] L. Xu, E. Oja, and P. Kultanen, “Randomized hough transform,” *Encyclopedia of Artificial Intelligence*, vol. 3, pp. 1354–1361, 2009. 2.1.2.2

[YH14] W. N. J. H. W. Yussof and M. S. Hitam, “Invariant gabor-based interest points detector under geometric transformation,” *Digital Signal Processing, DSP*, vol. 25, pp. 190–197, 2014. 2.1.2.2

[YITY05] Y. Yagi, K. Imai, K. Tsuji, and M. Yachida, “Iconic memory-based omnidirectional route panorama navigation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 27, no. 1, pp. 78–87, 2005. 3.3.2

[YO03] Y. U. Yim and S.-Y. Oh, “Three-feature based automatic lane detection algorithm (tfalda) for autonomous driving,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 4, no. 4, pp. 219–225, 2003. 3.1.2

[Zha00] Z. Zhang, “A flexible new technique for camera calibration,” *IEEE Transactions on Pattern Analysis and Machine Intelligence, (T-PAMI)*, vol. 22, no. 11, pp. 1330–1334, 2000. 2.1.1.2

[ZK09] A. M. Zhang and L. Kleeman, “Robust appearance based visual route following for navigation in large-scale outdoor environments,” *The International Journal of Robotics Research, (IJRR)*, vol. 28, no. 3, pp. 331–356, 2009. 5.1

[ZK11] L. Zhang and R. Koch, “Hand-held monocular SLAM based on line segments,” *Proceedings. International Machine Vision and Image Processing Conference, (IMVIP 2011)*, pp. 7–14, 2011. 2.1.2.4, 3.2.2, 3.3.1, 6.1

[ZK13] ——, “An efficient and robust line segment matching approach based on lbd descriptor and pairwise geometric consistency,” *Journal of Visual Communication and Image Representation*, vol. 24, no. 7, pp. 794–805, 2013. 2.8, 2.1.2.3, 2.1.2.4, 6.1, 6.2.1, 7.2

[ZWT03] C. Zhou, Y. Wei, and T. Tan, “Mobile robot self-localization based on global visual appearance features,” in *Proceedings. IEEE International Conference on Robotics and Automation, (ICRA 2003)*, vol. 1, 2003, pp. 1271–1276. 3.5.2

Résumé

CETTE thèse présente une méthode de navigation par asservissement visuel à l'aide d'une mémoire d'images. Le processus de navigation est issu d'informations d'images 2D sans utiliser aucune connaissance 3D. L'environnement est représenté par un ensemble d'images de référence avec chevauchements, qui sont automatiquement sélectionnés au cours d'une phase d'apprentissage préalable. Ces images de référence définissent le chemin à suivre au cours de la navigation. La commutation des images de référence au cours de la navigation est faite en comparant l'image acquise avec les images de référence à proximité. Basé sur les images actuelles et deux images de référence suivantes, la vitesse de rotation d'un robot mobile est calculée en vertu d'une loi du commandé par asservissement visuel basé image. Tout d'abord, nous avons utilisé l'image entière comme caractéristique, où l'information mutuelle entre les images de référence et la vue actuelle est exploitée. Ensuite, nous avons utilisé des segments de droite pour la navigation en intérieur, où nous avons montré que ces segments sont de meilleurs caractéristiques en environnement intérieur structuré. Enfin, nous avons combiné les segments de droite avec des points pour augmenter l'application de la méthode à une large gamme de scénarios d'intérieur pour des mouvements sans heurt. La navigation en temps réel avec un robot mobile équipé d'une caméra perspective embarquée a été réalisée. Les résultats obtenus confirment la viabilité de notre approche et vérifient qu'une cartographie et une localisation précise ne sont pas nécessaire pour une navigation intérieure utile.

Mots-clés Navigation visuelle, Asservissement visuel.

Abstract

THIS thesis presents a method for appearance-based navigation from an image memory by Image-Based Visual Servoing (IBVS). The entire navigation process is based on 2D image information without using any 3D information at all. The environment is represented by a set of reference images with overlapping landmarks, which are selected automatically during a prior learning phase. These reference images define the path to follow during the navigation. The switching of reference images during navigation is done by comparing the current acquired image with nearby reference images. Based on the current image and two succeeding key images, the rotational velocity of a mobile robot is computed under IBVS control law. First, we have used the entire image as a feature, where mutual information between reference images and the current view is exploited. Then, we have used line segments for the indoor navigation, where we have shown that line segments are better features for the structured indoor environment. Finally, we combined line segments with point-based features for increasing the application of the method to a wide range of indoor scenarios with smooth motion. Real-time navigation with a Pioneer 3DX equipped with an on-board perspective camera has been performed in indoor environment. The obtained results confirm the viability of our approach and verify that accurate mapping and localization are not mandatory for a useful indoor navigation.

Keywords Vision-based navigation, Visual Servoing.