Techniques for Extracting Contours and Merging Maps

A Dissertation
Submitted to
the Temple University Graduate Board

---

in Partial Fulfillment
of the Requirements for the Degree of
DOCTOR OF PHILOSOPHY

---

by
Nagesh Adluru
November 26, 2008

ABSTRACT

Techniques for Extracting Contours and Merging Maps

Nagesh Adluru
DOCTOR OF PHILOSOPHY

Temple University, November 26, 2008

Longin Jan Latecki, Chair

Understanding machine vision can certainly improve our understanding of artificial intelligence as vision happens to be one of the basic intellectual activities of living beings. Since the notion of computation unifies the concept of a machine, computer vision can be understood as an application of modern approaches for achieving artificial intelligence, like machine learning and cognitive psychology. Computer vision mainly involves processing of different types of sensor data resulting in "perception of machines". Perception of machines plays a very important role in several artificial intelligence applications with sensors. There are numerous practical situations where we acquire sensor data for e.g. from mobile robots, security cameras, service and recreational robots. Making sense of this sensor data is very important so that we have increased automation in using the data. Tools from image processing, shape analysis and probabilistic inferences i.e. learning theory form the artillery for current generation of computer vision researchers.

In my thesis I will address some of the most annoying components of two important open problems viz. object recognition and autonomous navigation that remain central in robotic, or in other words computational, intelligence. These problems are concerned with inducing computers, the abilities to recognize and navigate similar to those of humans. Object boundaries are very useful descriptors for recognizing objects. Extracting boundaries from real images has been a notoriously open problem for several decades in the vision community. In the first part I will present novel techniques for extracting object boundaries. The techniques are based on practically successful state-of-the-art Bayesian filtering framework, well founded geometric properties relating boundaries and skeletons and robust high-level shape analyses.

Acquiring global maps of the environments is crucial for robots to localize and be able to navigate autonomously. Though there has been a lot of progress in achieving autonomous mobility, for e.g. as in DARPA grand-challenges of 2005 and 2007, the mapping problem itself remains to be unsolved which is essential for robust autonomy in hard cases like rescue arenas and collaborative exploration. In the second part I will present techniques for merging maps acquired by multiple and single robots. We developed physics-based energy minimization techniques and also shape based techniques for scalable merging of maps. Our shape based techniques are a product of combining of high-level vision techniques that exploit similarities among maps and strong statistical methods that can handle uncertainties in Bayesian sense.

ACKNOWLEDGEMENTS

I have a relatively long story to tell as lot of people influenced in shaping my research abilities and scientific efforts and dreams. Besides having a dream to be a scientist I had no *concrete* idea of what I would do when I graduated back in India. I will start from most recent people tracing back to those in my college. When I accidentally ran into Longin Jan Latecki in his newly allocated ”vision lab” on the 10th floor, he said his main focus is *making sense of sensor data*. Longin's constant drive towards producing results in that direction has been the most influential part in my transformation into a computer vision researcher. I am deeply grateful for his support, patience and stimulation in helping me become productive in the field of computer vision. His passion to publish though overwhelming at times has definitively instilled similar passion in me for better. Now let me thank another mentor and friend, Rolf Lakämper for his soothing philosophies that helped me triumph over my anxieties in my struggle to survive in research. Needless to mention, I had very fruitful collaboration with him on projects of robot mapping. His versatility in advising is the rarest combination I found at Temple. I sincerely thank Slobodan Vućetić and Marc Sobel for serving on my committee. In applied research collaboration usually pays off for higher productivity. It has been very true in my case. I thank all the collaborators I have worked with (ChengEn Lu, Haibin Ling, Thomas Young, Xingwei Yang) for boosting my productivity. Another Temple affiliate who molded my initial years at Temple is my Masters adviser, Richard Beigel. In addition to providing unequivocal support, he allowed me to experiment with different research directions that led to a fruitful Masters project. His brilliance in complexity theory opened up gates for me to the theory community which guides me on many different levels.

Silly as it might sound, the writing style of the beginning of this thesis has been inspired by that of Scott Aaronson (MIT) whose views on Computer Science and life make him my hero. Above all, in this systemic world I thank American system for giving me *second* chances! Even though it is quite a far connection, Kumar Eswaran, my undergrad adviser, helped me realize the potential of using the field of Computer Science to satisfy my passion for discovery. My life at Temple was the gateway to my life in America: Amazing friendliness of Suzan Köknar-Tezel, Jon Ikoniac, Thomas Stauffer, Toni Newton and many others here helped me in keeping me focussed on my work during ups and downs. I thank all those whose encounters transformed me incredibly and for better. I sincerely feel fortunate for having a family that is patient, supportive and providing base for learning some of the most important lessons in life. For that I thank my mom, dad and brother.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Philosonomicon: Is vision part of intelligence?

1.1 Human vision, learning and intelligence

Charles Darwin in his *On the origin of species* said: ”the evolution of the eye by natural selection at first glance seemed ’absurd in the highest possible degree’. However despite the difficulty in *imagining* it, it was perfectly feasible...”, [56]. Studying eye is as much related to understanding vision as computers are to understanding computation. Even though a baby can capture images as soon as it is born, it takes time for the baby to actually start ”seeing” or ”perceiving”. It involves ability to learn. Nevertheless it seems that the learning algorithms they use in vision seem to be evolutionary heritage as they are so ”hard-wired” into the genes similar to our ”intelligence”. It’s like having all the required training data and and millions of years to learn and then build the program as an embedded circuit. Here human vision can be thought of not just ability to use visual information but any sensor information (like for e.g. blind people using their hands or blind sensing sticks) to perform activities like recognition, navigation and interaction. Hence human vision *is* part of our intelligence.

A crucial part in characterizing or defining human vision, learning or intelligence is *communication*. Without communication there’s no way to verify something in a scientific way. For communication we need material. Languages evolve and the more precise one can communicate the better will be the ability to learn, behave intelligently, lead the food chain and hence survive through odds. Since the beginning of enlightenment era around late 17^{th} century and early 18^{th} century ”reason” began to be the basis of authority. Hence the quest to understand human intelligence can be thought to have begun since then.

A hypothesis that is very plausible is that all actions are seeded in *thoughts*. Our understanding of human intelligence will be limited by our understanding of the language of thoughts.

Hence the pursuit of understanding the language of thoughts is well justified.

1.2 Artificial vision, learning and intelligence

Understanding Nature for engineering purposes has to do with what is observable, measurable and repeatable with certain level of predictability. Even though logic and math existed for centuries before the advent of computers, a new era of science was ushered by efforts of Alan Turing and Alonzo Church who came up with the famous Church-Turing thesis. Turing's famous paper *On computability* [256] essentially can be thought of as a heuristic explanation of the human thought process which seems to be very successful. Thus Turing proposed a language for the machine human interaction. The book [92] provides a nice introduction to the theory of computability. His famous test for Artificial Intelligence involves modeling thought process as a computational device and comparing it with *behavior* of a human. Even though computational model of human thought process is only a heuristic it nevertheless models many many human thought processes. With the advent of complexity theory it became apparent that computation may not *efficiently* model all thought processes as argued by works like Scott Aaronson's thesis [1].

But as Niels Bohr once said, *It is wrong to think that the task of physics is to find out how Nature is. Physics concerns what we say about Nature.* efforts in artificial intelligence also concern themselves with what we can *say* about intelligence not what it *is.* This lead to the modern approaches in machine learning. To summarize the efforts in one long sentence would be to say that those are essentially based on principle of Occam's razor to explain the unknown distributions of observed data using different efficient algorithms and models of data. Different sub-fields (like vision, bio-informatics, medical imaging, robotics etc.) essentially involve in coming up with those efficient algorithms and data modeling. For e.g. artificial vision deals itself with understanding data obtained using electromagnetic spectrum starting from X-rays (CT-scans) to visible light (photo cameras) to radio waves (MRI). Goals in artificial vision and intelligence thus don't necessarily *restrict* themselves to human abilities. Humans just form the lower bound of what we want to do with machines.

As long as $P \neq NP$ *we* have to design the learning algorithms for the machines to learn and behave. Understanding the way human brain really works may help to come up with better computational models that actually mimic the underlying process generating the human intelligence. But it's a long way to go. Right now Turing model is the most promising because of the enormous creative abilities of humans in designing clever algorithms.

1.3 Types of efforts in computer vision

In the grand scheme of pursuit of AI computer vision interacts with many other subfields as shown in Fig. 1.1. Computer vision can be thought of mining sensor data for useful information. The

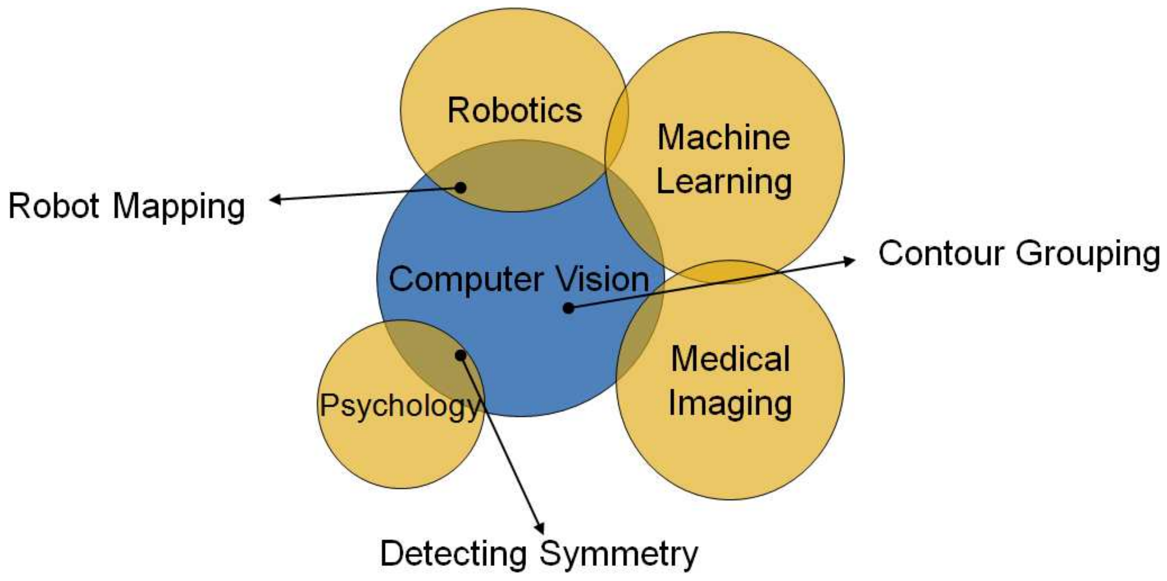


Figure 1.1: Computer Vision interacts with many other subfields like robotics, psychology, medical imaging, machine learning etc. The arrows show the areas I worked on in the process of getting my PhD.

sensor data can be from digital cameras that capture 2D images or videos, can be 3D data obtained from stereo or range sensors. The data can also be from non-visual electromagnetic spectrum like radio waves, X-rays which is mainly used in medical imaging. This spectrum helps to "see" the insides of human beings and things.

Mining sensor data involves understanding the properties of sensors, the characteristics of the data collected. The efforts range from modeling physics of sensors to extracting useful features of data and adapting machine learning tools.

For e.g. work like [291, 2] studies the reflectance and spectral properties of light towards rendering and understanding images. Work like [288, 289] pursues understanding the role of appearance information like color, texture etc. in computer vision. There is a lot of work done by psychologists to understand how humans try to combine different cues from images into meaningful objects ([76, 78, 81]). There are many different types of effort like shape analysis for object recognition ([234, 232, 231]), shape extraction by segmentation ([281, 53]), using text mining techniques like bags-of-features for object categorization etc. ([109]) and machine learning techniques ([259]).

Recognition and navigation are two central problems for intelligent robotics with many applications. For e.g. image *based* retrieval, video surveillance, medical robotics etc. Images can be retrieved by annotations but for robots to interact they need to recognize objects by processing the images directly. Contour grouping and robot mapping are two central components for recognition and navigation and this thesis presents my work in those two areas.

Part I

Contour Grouping

CHAPTER 2

Contour Grouping Based on Contour-Skeleton Duality

In this paper we present a method for grouping relevant object contours in edge maps by taking advantage of contour-skeleton duality. Regularizing contours and skeletons *simultaneously* allows us to to combine both low level perceptual constraints as well as higher level model constraints in a very effective way. The models are represented using paths in symmetry sets. Skeletons are treated as trajectories of an imaginary virtual robot in a discrete space of ”symmetric points” obtained from pairs of edge segments. Boundaries are then defined as the maps obtained by grouping the associated pairs of edge segments along the trajectories. Casting the grouping problem in this manner makes it similar to the problem of *Simultaneous Localization and Mapping (SLAM)*. Hence we adapt the state-of-the-art probabilistic framework namely Rao-Blackwellized particle filtering that has been successfully applied to SLAM. We use the framework to maximize the joint posterior over skeletons and contours.

2.1 Background

Object boundaries, also known as contours, are very useful descriptors for object recognition. Extracting contours using only edge-detection algorithms without any regularization is an ill-posed problem due to ambiguities in the gradient space of real-world images. Hence *grouping* edge pixels into contours also known as contour grouping is a very popular approach. The problem even after decades of work still remains to be open and a very active research field as documented by recent papers, [285, 253, 244, 124, 242, 100]. Different perceptual grouping constraints motivated by Gestalt psychology ([269]) and cues like closure, good-continuity ([194]), minimal model theory ([80, 77]) have been used to regularize contour growth. All such low-level grouping constraints can be used in extracting ”smooth and compact” contours but not necessarily those useful for object recognition problem. Hence researchers started using higher level constraints which when used appropriately can result in ”relevant contours”. This is possible because high level constraints can capture both local and global characteristics of an image similar to how humans process images. For example [124] uses ”depth cues” to recover *occluding* contours. [242] uses motion cues where the central idea is based on the fact that if the ”object of interest” moves around in a sequence of image frames, its contours can be detected by capturing the dynamics of the occluding contours. Their system is based on the belief that the situation of having a sequence of frames is more natural than using a static image for object recognition. But it is clear that even though humans might exploit the motion cues they do not *depend* on those to detect objects. Also it is not always possible to have a sequence of frames for the task of recognition.

The use of symmetry as a key contour grouping cue has been studied in both human vision and computer vision. Among others, the results in [181, 214, 175, 287, 142] show that symmetry is non-accidental. Therefore symmetry can be expected to be useful in not only distinguishing salient contour structures from noisy background in the low-level processing but also in extracting *more relevant* contours for the task of object recognition. Relevant contours are closely related to *shape* of an object. Shapes of objects and their symmetries have several dual geometric properties ([107, 152]). Shapes can be represented using signatures defined using locally symmetric curves ([155]) and shapes can be reconstructed from symmetries ([106]). Symmetry for rigid and non-rigid objects can be captured in several ways: for example using ribbons ([194]), using planar reflections ([212]), using intrinsic self-similarities ([216]) or more classically using medial axes ([27, 28]) and symmetry sets ([37, 105]). A lot of work has been done in using symmetry for object recognition ([233, 228, 151, 154, 14]). Symmetry principle expressed as *global* contour symmetry has been used in contour grouping in various approaches. One of the more recent approaches that is based on global contour symmetry is presented in [241]. It is related to the grouping method developed in [194], where symmetry is considered along with closure and proximity. Symmetry is applied as a cue to pair the extracted curves by producing a set of ribbons. These ribbons are then grouped into structures using heuristic algorithms. While different representations of symmetry have different advantages based on the context of application, those that can capture *local* symmetry play more important role in grouping contours, since global symmetry is usually not present in 2D images due to perspective distortion and nonrigid deformation.

Probabilistic reasoning is popular in processing noisy images. For e.g., a popular set of edge detectors, *pb*, is built using probabilistic representation of boundary ([188]). In contrast, our approach represents a joint-posterior of *both* contours and skeletons. Also a probabilistic method for tracking of motion boundaries for motion estimation is presented in [25]. Bayesian reasoning is gaining popularity even in psychophysical theory of vision. For example, [79] presents a Bayesian framework to probabilistically judge grouping hypotheses used by humans.

Bayesian approach for grouping contours using multiple hypotheses tracking was introduced in the seminal paper, [54] which is heavily based on another seminal work in [217]. More recently, a particle filter based system called JetStream was applied to contour grouping in [209]. This work treats the grouping process as a dynamic process and the detection is performed on edge pixels with particles following the contour directly. In this paper a simple ribbon geometry is also used for road extraction. They introduced a novel and an unconventional version of a particle filter tracking algorithm, where temporal sequence is replaced with a sequence of growing contour points. JetStream often fails to track contours in complex images, since only low level features (image gradient and corner detection) are used as the basis for particle observations. Therefore, it requires user interaction during the tracking process.

Since particle filtering (also known as sequential Monte Carlo estimation) provides a strong framework with Bayesian reasoning to capture complex (non-linear and non-gaussian) dynamics of probability density functions it has been extensively applied for robust object tracking. One of the best known approaches in computer vision is the Condensation algorithm ([133]), which allows tracking object contours in the presence of background clutter. Particle filtering has become the standard approach for mobile-robot localization with the main application being SLAM ([251, 117, 69]), where probability distributions for the robot poses (position plus heading direction) and the possible maps are approximated and propagated by a set of particles. We first cast contour grouping as SLAM problem in [3]. The presented work extends this idea in several ways as summarized in Section 2.2. Since we use a strong statistical framework with constraints that are not entirely low-level, our work has advantage of capturing relevant contours even in the presence of significant distractors and inner structures. While the techniques like [209] use good statistical framework they do not exploit geometric information. The methods like [179, 219] exploit geometry but are limited in noisy conditions.

2.2 Overview of our approach

Skeletons capture local symmetry in a very useful way. According to Blum's definition a skeleton S of a set of object boundaries D, is the locus of the centers of maximal disks. A maximal disk in D is a closed disk contained in D that is interiorly tangent to the boundary of D and that is not contained in any other disk in D. Each maximal disc must be tangent to the boundary in at least two different points. A set of skeleton points $s \in S$ and the radii $r(s)$ of their maximal disks can be used for reconstructing the boundary without any ambiguity. An important property is that skeletons can be computed for every planar shape, but computation of skeletons without boundary information is not possible. Thus, there is cyclic dependency between contours and skeletons, which is called contour-skeleton duality. The key idea of our approach is to exploit this dependency for grouping relevant contours by *simultaneous* estimation of skeletons and the contours in edge images. Thus we regularize both contours and skeletons. A detailed analysis of medial axis properties and algorithms is presented in a recent book, [235]. A good mathematical introduction about medial axis transform can also be found in [50]. Skeletons can be used to represents shape models effectively ([253]). This allows us to incorporate higher level model information effectively in a sequential way which significantly reduces the risk of accidental groupings of edge pixels. Since grouping is inherently a *sequential* process, we maximize a joint-posterior of contours and skeletons using Bayesian *filtering*. We adapt a practically very successful approach Rao-Blackwellized particle filtering which is used for the problem of Simultaneous Localization and Mapping (SLAM) in the field of robot mapping. We treat skeletons as trajectories of a virtual robot and the maps of associated edge segments as boundaries. The odometric and range constraints are replaced by perceptual grouping and model constraints. Perceptual constraints can be viewed as practical realization of Gestalt grouping principles ([269]).

Our work is related to the grouping method developed in [179] in that local symmetry axes are used. They identify segments along local symmetry-axis and apply a shortest-path algorithm to connect some of them into a complete symmetry axis. The grouping cost function is defined as the sum of local costs along the symmetry axis. In addition to using different measures, we have a more powerful computational framework in the proposed approach. However, the main difference is the usage of flexible shape models based on symmetry sets to guide contour grouping in our approach. Integer Quadratic Programming is is used in [219] to group contour segments based on constrained Delaunay triangulation. In contrast to our proposed approach, this approach fails in the presence of distractor edges induced by object inner structures. Moreover, grouping of only parts of contours is possible in our framework. This is also in contrast to active contour based methods ([26]).

Fig. 2.1 shows the algorithmic overview of the proposed approach. (1) For a given input image, an edge image is computed. (2) The edge pixels are then linked to form chains which are approximated as edge segments. We currently use publicly available code ([149]). Other sophisticated linking algorithms like [244, 285] could also be used. (3) The edge chains produced by low-level linking algorithms are often too long and run into noise and boundaries of different objects. Hence the edge chains are split into "scale adaptive edge segments". (4) Boundary and skeleton of the object of interest are grouped using the scale adaptive edge segments in a probabilistic framework. A reference shape model and perceptual grouping constraints from contour-skeleton duality are used as constraints. The grouping process is based on Rao-Blackwellized particle filtering framework. The first three steps can be considered as low-level preprocessing steps. The last step, which is our main contribution, exploits mid-level (contour-skeleton properties) and high-level (shape properties) of model objects.

This work is an extension of [3]. We made several improvements that make our system more robust and also work on even small scale images. The improvements are summarized below.

1. Design of proposal and importance weights are very crucial in using the particle filtering framework. A complex optimal proposal as introduced in [283] and presented in [117] was used in our previous work ([3]). Using optimal proposal gives us advantage when observation sensors are more accurate than motion sensors. More specifically when the modes in proposal

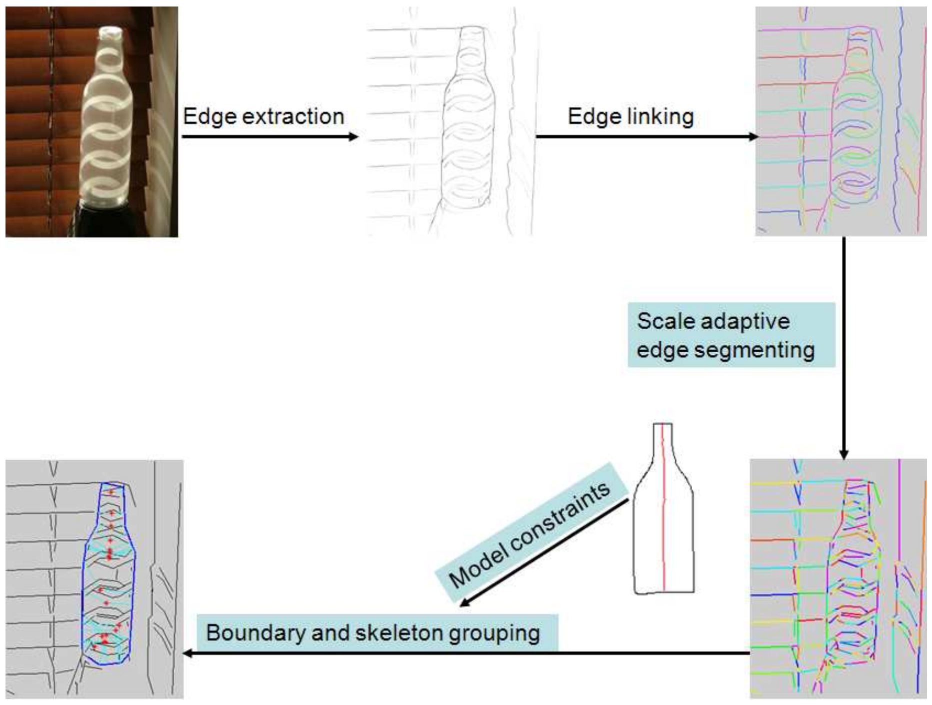

Edge extraction
Edge linking
Scale adaptive edge segmenting
Model constraints
Boundary and skeleton grouping

Figure 2.1: Overview of our algorithm: (1) Edge extraction. (2) Edge linking. (3) Approximation by scale adaptive edge segments . (4) Grouping boundary and skeleton simultaneously using Rao-Blackwellized particle filter with symmetry based model constraints.

function are very different from the modes in the likelihood function. Since we want to keep our models flexible, we do not treat the model sensor to be accurate. Therefore, in this paper we use "prior proposal" following [123, 122, 246]. The prior proposal captures the dynamic of our system namely contour and skeleton growth. The growth is regularized using constraints from contour-skeleton duality. Thus the proposal chooses smooth extensions for both contours and skeletons. While the model helps us distinguish smooth contours and skeletons from those of interest, we are not forced to follow the model constraints closely, thus improving model deformability. Also to handle the multiple modes in our proposal we use prior boosting ([108, 42]). The details of simulating this proposal are explained in Section 2.4.2.

2. We extend our shape model to be based on symmetry sets ([37]) instead of medial axes paths since symmetry sets capture local symmetry of even *partial* contours. We choose the longest path in the symmetry set that represents the significant pair of boundaries in the model shape. The model constraints encode not only the shape of the boundary pair but also the shape of the symmetric path itself. The details of these improvements are explained in Section 2.5.

3. Orientations of edge pixels play an important role in our framework. In [3], we used line fitting

based on EM [170] and sampled equidistant pixels from the segments with the orientations of the segments. These equidistant sampled pixels formed the set of edgels (edge pixels with orientations) which were then grouped. In the current system we exploit the low-level edge linking code of [149] and obtain ”scale adaptive edge segments” which are then grouped. The scale adaptive edge segmenting is similar in spirit to breaking of chains into linear segments as in [220]. In [220] chains are broken at the sharp curvatures. Here in addition to breaking at the sharp curvatures we break them based on surrounding edge distribution as explained in Section 2.6. We would like to note that our framework does not *depend* on edge-linking. It uses the low-level information if available. If no linking is available, we can just group edge pixels with orientations for example those given by third order gradient introduced in [244]. The better the low level edge linker is, the faster the virtual robot marches in the grouping process, because it groups longer edge segments. Thus, this is a system level improvement that is practically very relevant especially since lot of progress has been made in low-level linking, for example [244, 285]. The details of these implementations are explained in Section 2.6.

The rest of the paper is organized according to the algorithmic flow in our system. Section 2.3 describes our main framework in which the virtual robot groups the contours and skeletons using Rao-Blackwellized particle filtering. Section 2.4 describes the details related to simulation of our proposal based on perceptual constraints from contour-skeleton duality. Section 2.5 describes the evaluation of importance weights for the particles based on reference shape model constraints. The implementation details are presented in Section 2.6. Section 2.7 presents our experimental results and then finally conclusions and discussions are presented. We would like to note that a lot more experimental results and demonstrative videos are presented in our supplementary material submitted with this paper. Also we use the terms skeletons and symmetry axis interchangeably and our skeletons do not necessarily mean traditional medial axes.

2.3 Probabilistic grouping of contours and skeletons

In this section we relate simultaneous grouping of contour segments and skeleton points to the problem of simultaneous localization and mapping (SLAM). Then we describe the probabilistic framework used for our grouping task.

2.3.1 Mapping and localizing simultaneously

A robot needs to localize itself in an environment for autonomous navigation. For a robot to localize using local sensors, it is important to have a map of the environment. Typically the maps are not available up front and might be quite inaccurate so the robot has to build the map of the environment by itself. There is a cyclic dependency between the tasks of mapping and localizing: mapping requires knowing the position of the robot, and robot’s position can only be recorded in a map. Since the sensors are usually noisy, probabilistic approaches are needed and have been successfully applied for the problem. For a comprehensive survey see [251]. In such probabilistic approaches a joint-posterior representing trajectory, $x_{1:t}$, of the robot and map, m_t of the environment is maximized. A sequence of sensor measurements namely range measurements, $z_{1:t}$, and odometry readings, $u_{1:t}$, are used as constraints. The goal is to find:

$$\operatorname*{argmax}_{x_{1:t}, m_t} p(x_{1:t}, m_t | z_{1:t}, u_{1:t}) \tag{2.1}$$

Since both the trajectory and map are estimated simultaneously the problem is called *Simultaneous Localization and Mapping (SLAM)*. To allow for online optimization, *sequential* Monte Carlo estimation called *particle filters* have been successfully applied. The posterior is represented using a set of particles (random samples drawn from the posterior). Each particle, i, represents a trajectory $x^{(i)}_{1:t}$ up to time t and an associated map $m^{(i)}_t$ constructed up to time t. The dimensionality of the state being estimated plays a crucial role in efficient application of particle filters. Rao-Blackwellization allows to reduce the dimensionality by factorizing the states that are conditionally independent and can be estimated analytically. Since a map m_t can be analytically constructed for a given sequence of poses $x_{1:t}$ and observations $z_{1:t}$ ([200]), Rao-Blackwellization of the joint posterior results in

$$p(x_{1:t}, m_t | z_{1:t}, u_{1:t}) = p(m_t | x_{1:t}, z_{1:t}) p(x_{1:t} | z_{1:t}, u_{1:t}) \tag{2.2}$$

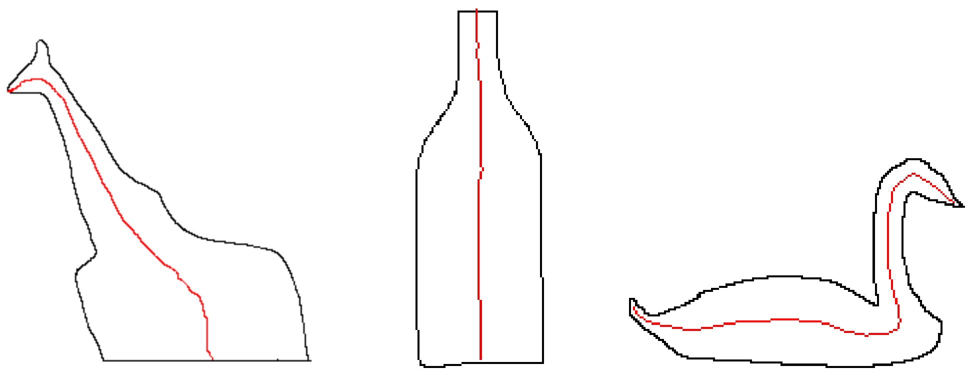
Once the state is decomposed any standard particle filter can be used, the most popular being Sampling Importance Resampling (SIR) filter ([117]). At every time step t, usually the most likely particle, i.e., the i^{th} particle that maximizes $p(x^{(i)}_{1:t}, m^{(i)}_t | z_{1:t}, u_{1:t})$ is used for localization and navigation.

2.3.2 Grouping contours and skeletons simultaneously

Our key idea is to maximize a *joint* posterior representing contours and skeletons. Intuitively we are trying to find the MAP (Maximum A Posteriori) estimate of:

$$p(\text{SKELETON, CONTOUR} \mid \text{MODEL, GROUPING CONSTRAINTS}) \tag{2.3}$$

Bayesian estimation of skeletons given a complete contour i.e. $p(\text{SKELETON}|\text{CONTOUR})$ was presented in [82]. In our case we estimate both skeleton and contour simultaneously based on model and perceptual constraints from contour-skeleton duality. The model constraints are the high level constraints that actually help our robot distinguish the contours of object of interest from distractor contours that are "perceptually smooth" but do not belong to the boundary of object of interest. Our framework permits effective way of using model constraints for contour grouping. Our shape model is based on the symmetry set ([37]) which can capture symmetry of even partial contours. Since the space of our symmetric points is a superset of the symmetry set (Section 2.4.1), by extending the model to be based on symmetry sets (SPs) we can capture objects with partial contours. To keep



(a) Giraffe (b) Bottle (c) Swan

Figure 2.2: The longest symmetric paths in the images form the bases of our shape models. The paths are subsets of the symmetry sets of the respective shapes. We choose the longest ones, since they capture the most significant boundary parts.

our model simple we choose the longest path in symmetry set that captures the significant boundary parts of the model, see Fig. 2.2. Thus, we replace the set of medial axes paths used in [3] with a single path in the symmetry set. The details of the path information are explained in Section 2.5.

For ease of understanding the optimization problem, we cast the grouping problem as the mapping problem of an imaginary virtual robot. The virtual robot walks in the space of locally symmetric points induced by pairs of edge segments. The construction of SPs, which generalizes medial axis points, is described in Section 2.4.1. The robot's trajectory composed of SPs represents a path in the space of locally symmetric points, $\mathbf{x}_{1:t}$ (SKELETON). The edge segments associated with the SPs form the contour map, c_t (CONTOUR). The robot's sensor information, Z_m, U_g (MODEL AND GROUPING CONSTRAINTS), is obtained from the reference shape model and grouping constraints based on contour-skeleton duality (Section 2.4.2). The goal is to maximize the joint posterior over the symmetric paths and contour maps:

$$\underset{\mathbf{x}_{1:t}, c_t}{\operatorname{argmax}}\; p(\mathbf{x}_{1:t}, c_t | Z_m, U_g). \tag{2.4}$$

Even though there are no explicit underlying dynamics based on real time, the grouping process is dynamic in the sense that the contours and skeletons *grow* in each step. Thus the dynamic system underlying the process is the "exploration". By inducing such virtual temporal information using the order of skeleton points ($\mathbf{x}_{1:t}$), we can produce *partial* skeletons and partial contours in case of occlusions of the objects of interest. The idea of inducing virtual temporal information for the grouping task is introduced in the seminal papers, [209, 54] which laid the foundation for Bayesian reasoning in the task of grouping. Since the contour c_t is conditionally independent and can be analytically computed given the sequence of skeleton points, $\mathbf{x}_{1:t}$ by grouping the corresponding contour segments of the center point ([28]), Rao-Blackwellization gives us the following equation in our case:

$$p(\mathbf{x}_{1:t}, c_t|Z_m, U_g) = p(c_t|\mathbf{x}_{1:t})p(\mathbf{x}_{1:t}|Z_m, U_g) \tag{2.5}$$

In the remainder of this section we provide the details of the Sampling Importance Resampling (SIR) particle filter for maximizing this posterior. Our goal is to find MAP estimate of the posterior $p(\mathbf{x}_{1:t}^{(i)}, c_t^{(i)}|Z_m, U_g)$. Thus a contour grouping decision is possible at every time step t. The contour $c_t^{(i)}$ and the skeleton $\mathbf{x}_{1:t}^{(i)}$ are determined by the most likely particles in the modes of the posterior $p(\mathbf{x}_{1:t}^{(i)}, c_t^{(i)}|Z_m, U_g)$. Clearly, at early grouping stages (for small t), we obtain only part of the contour and part of the skeleton of the object of interest. In each iteration, i.e., at every time step t, the following four steps are executed:

1) **Sampling/Proposal:** The next generation of particles $\{\mathbf{x}_{1:t}^{(i)}\}$ is obtained from the current generation $\{\mathbf{x}_{1:t-1}^{(i)}\}$ by sampling from a proposal distribution $\pi(\mathbf{x}_{1:t}|Z_m, U_g)$ which is assumed to satisfy the following recursion:

$$\pi(\mathbf{x}_{1:t}|Z_m, U_g) = \pi(\mathbf{x}_t|\mathbf{x}_{1:t-1}, Z_m, U_g)\pi(\mathbf{x}_{1:t-1}|Z_m, U_g) \tag{2.6}$$

Therefore, each particle is extended as $\mathbf{x}_{1:t}^{(i)} =< \mathbf{x}_t, \mathbf{x}_{1:t-1}^{(i)} >$ where $\mathbf{x}_t \sim \pi(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, Z_m, U_g)$.

2) **Importance weighting/Evaluation:** Since it is usually hard to design $\pi(\mathbf{x}_{1:t}|Z_m, U_g)$ that exactly simulates the true posterior $p(\mathbf{x}_{1:t}|Z_m, U_g)$. An individual importance weight $w(\mathbf{x}_{1:t}^{(i)})$ is assigned to each particle, according to:

$$w(\mathbf{x}_{1:t}^{(i)}) = \frac{p(\mathbf{x}_{1:t}^{(i)}|Z_m, U_g)}{\pi(\mathbf{x}_{1:t}^{(i)}|Z_m, U_g)} \tag{2.7}$$

The weights $w(\mathbf{x}_{1:t}^{(i)})$ account for the fact that the proposal distribution π in general is not equal to the true distribution of successor states. Under 1st order Markovian assumption and conditional independence the weights can be recursively estimated as:

$$\begin{aligned} w(\mathbf{x}_{1:t}^{(i)}) &= \frac{p(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, Z_m, U_g)\; p(\mathbf{x}_{1:t-1}^{(i)}|Z_m, U_g)}{\pi(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, Z_m, U_g)\; \pi(\mathbf{x}_{1:t-1}^{(i)}|Z_m, U_g)} \\ &= w(\mathbf{x}_{1:t-1}^{(i)}) \frac{p(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, Z_m, U_g)}{\pi(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, Z_m, U_g)} \\ &\propto w(\mathbf{x}_{1:t-1}^{(i)}) \frac{p(Z_m|\mathbf{x}_{1:t-1}^{(i)}, \mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, U_g)}{\pi(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, Z_m, U_g)} \text{ (using Bayes rule)} \end{aligned} \tag{2.8}$$

The proportionality in Eq. (2.8) is from normalization constant in Bayesian decomposition of $p(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, Z_m, U_g)$.

The choice of proposal π is a very important design criterion for successful implementation of a particle filter. The closer it is to true posterior the better the filter converges, with finite number of particles. In [3] we used a complex optimal proposal according to [63]. This was originally introduced in [283]. The basic idea of the optimal proposal is to use the model constraints (in our case Z_m) in designing π. We simulated the optimal proposal by approximating it with a Gaussian similar to the technique described in [117]. However, integrating Z_m into the proposal restricts the model deformability, because the particles are forced to closely represent the model. Therefore, we no longer use Z_m in the proposal. Instead we use the prior distribution, $p(\mathbf{x}_t|\mathbf{x}^{(i)}_{1:t-1}, U_g)$ as our proposal, following [123, 122, 246]. This proposal captures the virtual dynamic underlying our system i.e. "growth" of contours and skeletons. Since we prefer smooth contours and skeletons the prior distribution is simulated using grouping constraints, U_g from contour-skeleton duality. Hence in our case $\pi(\mathbf{x}_t|\mathbf{x}^{(i)}_{1:t-1}, Z_m, U_g) = p(\mathbf{x}_t|\mathbf{x}^{(i)}_{1:t-1}, U_g)$. Using this simple exploration based proposal the weight recursion in (2.8) becomes:

$$\begin{aligned} w(\mathbf{x}^{(i)}_{1:t}) &\propto w(\mathbf{x}^{(i)}_{1:t-1}) \frac{p(Z_m|\mathbf{x}^{(i)}_{1:t-1}, \mathbf{x}_t)\cancel{p(\mathbf{x}_t|\mathbf{x}^{(i)}_{1:t-1}, U_g)}}{\cancel{p(\mathbf{x}_t|\mathbf{x}^{(i)}_{1:t-1}, U_g)}} \\ &= w(\mathbf{x}^{(i)}_{1:t-1}) p(Z_m|\mathbf{x}^{(i)}_{1:t-1}, \mathbf{x}_t) \end{aligned} \tag{2.9}$$

$p(Z_m|\mathbf{x}^{(i)}_{1:t-1}, \mathbf{x}_t)$ represents particle evaluation with respect to the shape model. The details of computing this likelihood are explained in Section 2.5. The details of simulating the proposal are explained in Section 2.4. Intuitively, our proposal is "exploration based" and we explore all SPs in a certain region of interest around $\mathbf{x}^{(i)}_{t-1}$ and assign them probability masses based on the "quality of continuation" determined using contour-skeleton duality constraints.

One important point to note is that *all* the posteriors in our case are *discrete* since the space of SPs is discrete.

3) **Resampling:** For the particle filter to converge the variance of the weights has to be low since ideally the particles are supposed to represent *random* samples. To avoid the problem of "weight degeneracy" ([145]) particles with low importance weights are replaced by those with higher weights. This step is necessary since only a finite number of particles are used. Resampling is a key step that allows application of a particle filter in situations in which the true distribution differs from the proposal. We employ residual resampling ([178]). Since resampling duplicates particles with higher weights there is a risk of "particle depletion". This occurs when all particles become identical which again means they are not random samples. To mitigate this effect an adaptive resampling schedule as proposed in [3] is used. The schedule is based on the measure $N_{eff}(= \frac{1}{\sum_{1=1}^{N_p}(w^{(i)})^2})$ introduced in [177], where N_p is the number of particles. This measure is related to the dispersion of weights and following [64, 117], we resample only if $N_{eff} < N_p/2$.

4) **Updating contour:** This step involves computing $p(c^{(i)}_t|\mathbf{x}^{(i)}_{1:t})$. The contour is represented as a discrete probability distribution of edge segments similar to occupancy maps ([200]). If an edge segment belongs to the contour its probability is one, otherwise it is zero. This is analytically computed because a contour can be reconstructed given a skeleton ([28]). At each step it essentially involves connecting the edge segments in the image closest to those that are involved in generating $\mathbf{x}^{(i)}_{1:t}$. Since we store c_t per particle it essentially involves just appending the new edge segments to $c^{(i)}_{t-1}$. The edge segments grouped upto time t will have the probability one, while all other edge

segments will have a zero probability of belonging to the contour. This is the step that allows us to ”Rao-Blackwellize” the joint posterior.

2.4 Proposal based on regions of interest

In this section we explain the details of simulating our proposal which we selected to be equal to the prior distribution, $p(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, U_g)$. Simulation of the prior distribution is based on the virtual dynamic of our system namely contour and skeleton growth. We first explain the space of locally symmetric points in which our robot walks and then explain the region of interest based simulation of the prior distribution.

2.4.1 Symmetric points

Given a pair of edge pixels $(\widehat{e}, \widetilde{e})$ with orientations, called edgels, a symmetric point (SP) $\mathbf{x} = (x, y)$ is computed as the center of the projections of the edge normals onto the bisector as shown in Fig. 2.3(a). The SP is a generalization of the center of maximal disk in the definition of medial axis. By this construction an SP $\mathbf{x}$ is characterized by

1. Distances $\widehat{r} = ||\widehat{e} - \mathbf{x}||$ and $\widetilde{r} = ||\widetilde{e} - \mathbf{x}||$. Since we want the SP $\mathbf{x}$ to remain a center of a disk, we require that $\widehat{r} = \widetilde{r}$ (with some very small tolerance).
2. Deviation angles $\widehat{\delta}$ and $\widetilde{\delta}$ are defined as the angles between the normal at $\widehat{e}$ and vector $\overrightarrow{\widehat{e}\mathbf{x}}$ and that between the normal at $\widetilde{e}$ and $\overrightarrow{\widetilde{e}\mathbf{x}}$ respectively.

If the deviation angles $\widehat{\delta} = \widetilde{\delta} = 0°$, the disk with center (x, y) and radius $r = \widehat{r} = \widetilde{r}$ is tangential to both edgels, see Fig. 2.3(b). However, we cannot require the deviation angles to be zero due to inaccuracies in the directions of edgels and the effects of discretization. Hence we must tolerate SPs for which $\widehat{\delta}, \widetilde{\delta} > 0°$, see Fig. 2.3(c). The smaller the angles the more tangential is the disk. Hence we measure the quality of an SP by the tangentiality of the disk. It is computed as:

$$\mathcal{N}(\delta, 0, S_\sigma), \text{ where } \delta = \frac{\widehat{\delta} + \widetilde{\delta}}{2} \tag{2.10}$$

where $\mathcal{N}(x, \mu, \sigma)$ is the Gaussian function with mean μ and standard deviation σ evaluated at x and S_σ is the tolerance parameter for skeleton regularization.

The space of SPs is related to the space of symmetry set (SS) and evolute of the object of a given contour. We obtain that the set of SPs is equal to SS if $\widehat{\delta} = \widetilde{\delta} = 0°$, since SS is the locus of centers of disks that are tangential in at least two distinct points on the contour ([37]) i.e., the requirement of maximal disks is dropped. If we allow both $\widehat{\delta}, \widetilde{\delta}$ to be larger than $0°$, the set of SPs is a superset of the union of SS and the evolute. For example in Fig. 2.4(c) we show SPs where the average of $\widehat{\delta}$ and $\widetilde{\delta}$ is $\leq 1°$. Evolute is the locus of the centers of curvature, i.e. centers of disks that osculate the object boundary ([180]). Fig. 2.4(b) shows the MA, SS and evolute of an ellipse[1] and

[1]Figure taken from Wikipedia: http://en.wikipedia.org/wiki/

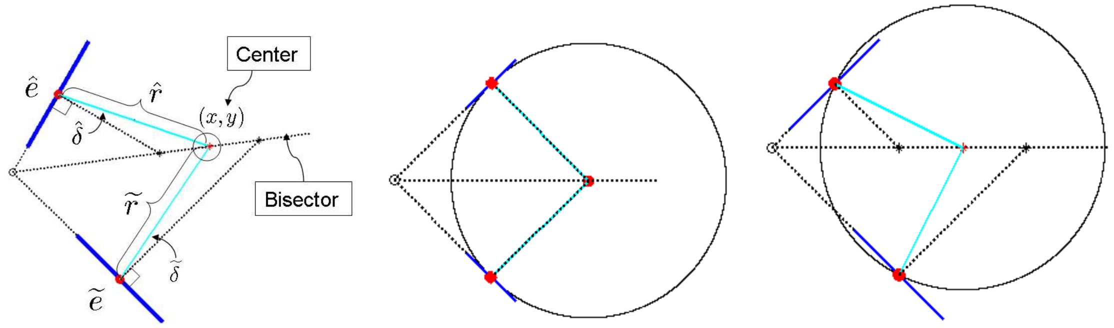


Figure 2.3: (a) Construction and components of a symmetric point. (b) An SP with circle touching tangentially to the edgels. (c) To tolerate the inaccuracies in the edge orientations and the effects of discretization we allow SPs with approximately tangential circles.

Fig. 2.4(a) shows MA and SS of a rectangle.

Even though MA completely represents a shape it cannot be used to capture symmetries of partial shapes. SS forms a superset of MA because it is similar to MA except that it does not require the circles to be maximally inscribing. The main difference between the space of SPs and SS and evolute is that we do not need the knowledge of true object boundary in the construction of SPs. [153] presents a technique for extracting skeletons using the symmetry set when boundaries are clearly defined. In contrast we try to extract skeletons and boundary simultaneously using the set of SPs.

2.4.2 Simulation of prior distribution

Now we explain the details of simulating our proposal distribution, $p(\mathbf{x}_t|\mathbf{x}^{(i)}_{1:t-1}, U_g)$. There are two main steps in simulating our proposal namely defining the region of interest for $\mathbf{x}_t$ and assigning probability masses to the center points in that region of interest. The region of interest is based on $\mathbf{x}^{(i)}_{1:t-1}$ and the probability masses are computed according to the grouping constraints from contour-skeleton duality, U_g. Thus we obtain a discrete probability distribution approximating $p(\mathbf{x}_t|\mathbf{x}^{(i)}_{1:t-1}, U_g)$ from which we sample the followers. Note that this discrete distribution is computed per particle.

The first step of designing region of interest is motivated by the recent work in *active* SLAM ([239, 47]). The basic idea in *active* SLAM is to plan robot's action to optimize its exploration of the environment by choosing from a set of actions weighed by the information gain. Since the goal of our virtual robot is to group the contour and skeleton of the object of interest we distinguish explored and unexplored regions and make the robot move into unexplored regions of interest.

Using $\mathbf{x}^{(i)}_{1:t-1}$ we can compute the explored region which is the interior of the contour, $c^{(i)}_{t-1}$

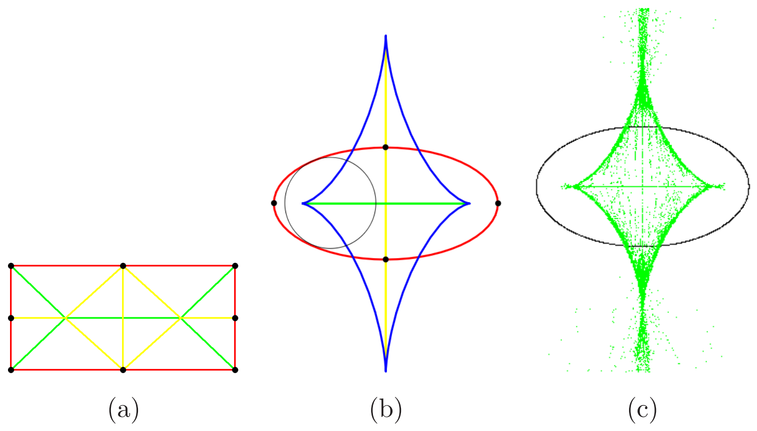


Figure 2.4: (a) Medial axis (green line), symmetry set (green and yellow) for a rectangle (red). The evolute for the rectangle is just at the corners since the boundary has curvature only at the corners. (b) Medial axis (green line), symmetry set (green and yellow lines) and evolute (blue curve) for an ellipse (red). (c) The space of symmetric points for an ellipse with the deviation angle threshold of $1°$.

grouped up to time $t-1$. Since we group pairs of contours along with skeleton points the interior of the boundary is known. We use a circular neighborhood of the endpoints of edge segments in unexplored regions (i.e., regions that are not enclosed by $c_{t-1}^{(i)}$) for possible extensions to the boundary. This can be seen in Fig. 2.5(a). The edge segments in the look ahead region are paired and then some pairs are culled in similar spirit to the culling used in DP-SLAM ([70]). We cull the pairs that extend boundary or skeleton with self-intersections, those that can not be used for inducing an SP, and those that induce an SP with radius out of range of the model radii. After culling, each pair of edge segments induces a possible follower SP, $\mathbf{x}_t$. Thus, at each step the virtual robot tries to extend the skeleton by expanding the contour. The process of looking ahead and the discrete space of $\mathbf{x}_t$s are shown in Fig. 2.5. The $\mathbf{x}_t$s in the region of interest are assigned probability masses according to $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, U_g)$. Intuitively this means computing the likelihood of $\mathbf{x}_t$ being a follower of $\mathbf{x}_{t-1}^{(i)}$ given the perceptual grouping constraints U_g. The perceptual grouping constraints, U_g can be understood as a practical realization of the Gestalt psychology principles in perception ([270]). In comparison to typical grouping approaches where a single contour is grown, we grow a pair of contours and their symmetry axis *simultaneously*. Hence our perceptual grouping constraints are motivated from contour-skeleton duality.

The skeleton growth constraint prefers robot's trajectory along good symmetric points (SPs). The quality of a symmetric point, $\mathbf{x}$ is measured as the average of its deviation angles, $\widehat{\delta}, \widetilde{\delta}$ as shown in Fig. 2.3(a). The smaller the average deviation the higher the quality of the SP. Fig. 2.3(b, c) show examples of a tangential (good) and a nearly tangential (not so good) SP. Both SPs are

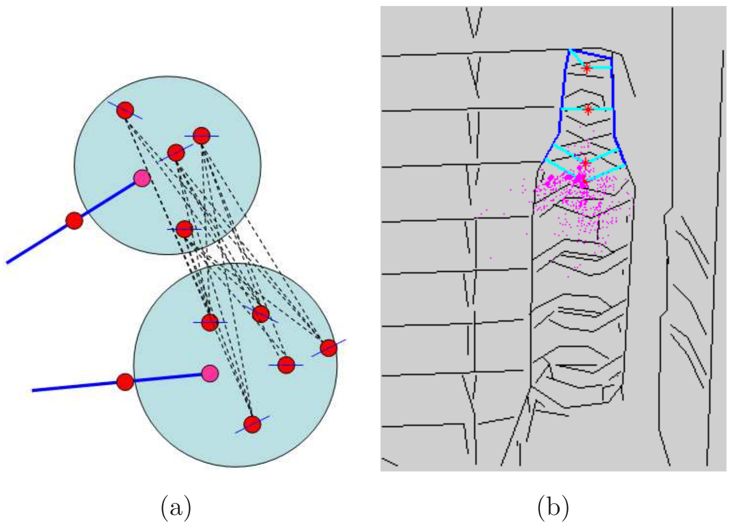


Figure 2.5: (a) The circular look ahead regions for possible contour extensions are shown as cyan circles. The possible extending edge segments in these regions are shown as red dots along with directions shown with short lines. All possible pairings of these edge segments generate the discrete space of $\mathbf{x}_t$s. (b) The contour grouped up to time $t-1$ i.e. $c_{t-1}^{(i)}$, is shown in blue. The region enclosed by this contour is designated as explored region for the virtual robot. We do not allow the look ahead segments to be in the explored region. The discrete space of $\mathbf{x}_t$s is shown with magenta points. The size of the dots is proportional to the probability mass computed according to $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, U_g)$.

allowed but the SP in (c) is of lower quality than that in (b). It is not possible for the robot to always be able to walk along good symmetric points because of noise in the boundary orientations and gaps in the boundary. The gaps are present because of discretization and missing parts of true edges. Hence we need a tolerance measure that restricts the drop in quality of SP. Since we regularize the skeleton *growth* the likelihood is computed based on the *change* in deviation angles of consecutive SPs. Therefore, we only penalize the drop in the quality of SP, $\mathbf{x}_t$ from that of the previous SP, $\mathbf{x}_{t-1}^{(i)}$. The likelihood of $\mathbf{x}_t$ as a follower SP according to this constraint is computed as $\mathcal{N}(s_t, 0, S_\sigma)$ which is a function of $s_t = \max(\delta_t - \delta_{t-1}^{(i)}, 0)$. δ_t and $\delta_{t-1}^{(i)}$ are the average deviation angles of $\mathbf{x}_t$ and $\mathbf{x}_{t-1}^{(i)}$ respectively. S_σ represents the tolerance for the constraint violation.

The contour growth constraint favors boundaries with smaller gaps. Let $g_t = (\widehat{g} + \widetilde{g})/2$ be the average gap between the edge segments of $\mathbf{x}_{t-1}^{(i)}$ and those of $\mathbf{x}_t$ as shown in Fig. 2.6. We use Gaussian $\mathcal{N}(g_t, 0, G_\sigma)$ to express the likelihood of $\mathbf{x}_t$ according to this constraint. Since contour gaps are unavoidable in real images because of missing gradient information, we have a certain tolerance of G_σ units in terms of pixels or sub-pixels.

Using the above perceptually motivated constraints, we compute the probability mass of

$\mathbf{x}_t$ as:

$$p(\mathbf{x}_t|\mathbf{x}^{(i)}_{1:t-1}, U_g) = \mathcal{N}(g_t, 0, G_\sigma) \cdot \mathcal{N}(s_t, 0, S_\sigma). \tag{2.11}$$

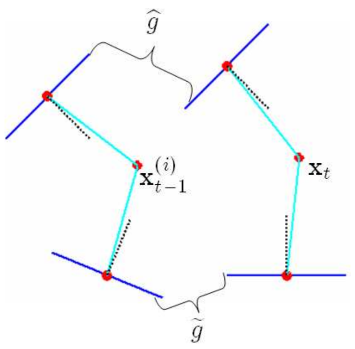


Figure 2.6: The average gap $g_t = (\widehat{g} + \widetilde{g})/2$ introduced by appending edge segments of SP $\mathbf{x}_t$ to those of SP $\mathbf{x}^{(i)}_{t-1}$.

Usually, the discrete distribution of $p(\mathbf{x}_t|\mathbf{x}^{(i)}_{1:t-1}, U_g)$ is multi-modal because of distractor segments and interior structures. It is important to have particles representing all meaningful regions of the posterior to be able to recover from distractions by noise. We use ”prior boosting” ([108, 42]) so as to capture multi-modal likelihood regions. In prior boosting more than one follower is sampled from $p(\mathbf{x}_t|\mathbf{x}^{(i)}_{1:t-1}, U_g)$, for each particle so that different followers can capture different modes of the likelihood of the posterior. The number of followers sampled typically depends on the number of modes which in turn depends on the local geometric conditions: if there are many distractor segments, then lot of followers are sampled and if there are less distractor segments then fewer followers are sampled. Thus after each iteration of Sampling Importance Resampling we might have more number of particles $N'_p > N_p$ but we retain only N_p *with their weights.* An important difference between resampling a fixed number of particles from large number of particles is that we retain their weights while after the adaptive resampling step the weights of all the particles are set to be equal.

An important parameter in the proposal is the size of the circular neighborhood L, which decides how far the virtual robot can ”look ahead” to define the region of interest that can capture the likelihood regions of the posterior. If the robot looks too far ahead, then it will be less sure if the boundary and skeleton extensions are true. On the other hand if it does not look far enough it might be distracted by noise and gaps in the true contour. The details of choosing the value of this and other parameters are explained in Section 2.6.

2.5 Evaluation based on shape model constraints

The importance weights for the particles are computed recursively according to Eq. (2.9), where $p(Z_m|\mathbf{x}^{(i)}_{1:t-1}, \mathbf{x}_t)$ measures how likely the particle $\mathbf{x}^{(i)}_{1:t}$ is according to the sensor information

Z_m. In this section we explain the sensor modeling for our virtual robot, which is based on the constraints from shape model.

As described in Section 2.3.2 our shape model is composed of the longest path in its symmetry set. The path is represented using a sequence of radii of the disks and *displacement vectors* of sample points on the path. The sequence of radii captures the shape of the contour generating the skeleton path, while the sequence of displacement vectors captures the shape of the skeleton path. Thus Z_m captures the shapes of both contour and major symmetric axis of the model image. Formally, if there are N sample points, $Z_m =< \{\overrightarrow{D_1}, R_1\}, \{\overrightarrow{D_2}, R_2\}, \ldots, \{\overrightarrow{D_N}, R_N\} >$, where $\overrightarrow{D_k}, R_k$ are the displacement vector and the radius of the k^{th} sample point respectively. A sample swan model is illustrated in Fig. 2.7.

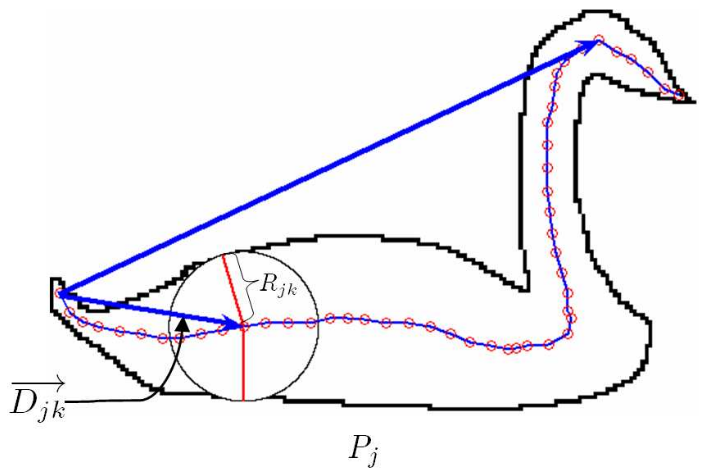


Figure 2.7: The swan model has one symmetric path: from tail to the beak. The path is shown in blue. The sample points are shown as red circles. Displacement vectors of two sample points are shown as blue arrows and the radius of one of the sample points is shown using the maximal disk.

Even though we have the *sequence* information for the path, the robot has *all* the information in each iteration. Hence this model is "static". At every time step (iteration), the robot computes its displacement vector since it knows its starting pose. Then it finds the closest displacement vector in the model and uses the expected radius associated with that sample point. Thus, the robot uses global information of the shape model to make a local decision in the grouping process. Formally, if the robot's current displacement vector is $\overrightarrow{(\mathbf{x}_1, \mathbf{x}_t)}$ which is a function of $\mathbf{x}_t$ and $\mathbf{x}_1$ (the starting pose), we first obtain:

$$\hat{k} = \underset{k}{\operatorname{argmin}}\; d_s(\overrightarrow{D_k} - \overrightarrow{(\mathbf{x}_1, \mathbf{x}_t)}), \quad k = \{1 \ldots N\} \tag{2.12}$$

where d_s measures the dissimilarity between two vectors, i.e., the distance between their orientations and magnitudes. Then using the index $\hat{k}$ we obtain the expected radius, $\mathcal{R}(\mathbf{x}_1, \mathbf{x}_t) = R_{\hat{k}}$ and the deviation from the closest displacement vector, $\mathcal{D}(\mathbf{x}_1, \mathbf{x}_t) = d_s(\overrightarrow{D_{\hat{k}}} - \overrightarrow{(\mathbf{x}_1, \mathbf{x}_t)})$. The values $\mathcal{R}(\mathbf{x}_1, \mathbf{x}_t)$ and $\mathcal{D}(\mathbf{x}_1, \mathbf{x}_t)$ constitute the sensor readings from the shape model for the robot at $\mathbf{x}_t$, whose trajectory started at $\mathbf{x}_1$. Using these sensor readings we compute

$$p(Z_m|\mathbf{x}_{1:t-1}^{(i)}, \mathbf{x}_t) = 0.5 \cdot \mathcal{N}(\mathcal{D}(\mathbf{x}_1^{(i)}, \mathbf{x}_t), 0, D_\sigma) + 0.5 \cdot \mathcal{N}(r(\mathbf{x}_t), \mathcal{R}(\mathbf{x}_1^{(i)}, \mathbf{x}_t), R_\sigma) \tag{2.13}$$

where D_σ and R_σ are the tolerance parameters for deformability in shape of the skeleton path and shape of the contour from model, respectively. The use of displacement vectors to obtain the expected radius at each step requires that one of the ends of the shape is present in the image. This is not a serious limitation because we can have two-way path for each model. The intuition behind this reasoning is that if neither ends of the path is present then the shape itself might be not be recognizable. Different parts of the object have different saliencies. Usually the more salient parts can be expected to be associated with the ends of symmetric paths.

The main advantages from the model proposed in this paper over the model in [3] are:

- By using the longest symmetric path as the model, we can capture the most significant pair of contours even if the complete contour is not available.
- By using the path information in a static way the robot uses global information from the model to make local decisions. This allows us to ignore minor shape details which can be easily confused with noise in real images.
- By using displacement vectors to obtain the expected radius at a time step (instead of using a sequence of radii), we have decoupled the parameters for controlling model deformability in terms of contour and skeleton. The deformability is captured by the deformability in the trajectory of the robot. The larger the tolerance D_σ for $\mathcal{D}(\mathbf{x}_1^{(i)}, \mathbf{x}_t)$ is, the more deformable the trajectory (shape of skeleton) can be. The larger the R_σ i.e. tolerance for deviation from $r(\mathbf{x}_t)$ is, the more deformable the contour can be.
- The use of displacement vectors allows us to group larger pieces of contour segments by exploiting low-level linking algorithms without loosening model constraints.

2.6 Implementation details

In this section we explain the details of using the low-level edge linking to reduce the space of SPs for faster grouping and the values of the parameters used. To take advantage of the progress made in low-level linking of edge pixels we first obtain edge chains. We use publicly available code, [149] but any other low-level linker can be used, for e.g. [285]. The edge linking process generally produces smooth chains of edge pixels as shown in Fig. 2.8(a), but the edge chains often are too long and run into noise. Hence we split these chains into small linear segments adaptively based on the distribution of surrounding chains and then perform grouping on these small linear segments which we call *scale adaptive edge segments*. The scale adaptive edge segmenting is similar in spirit to the curvature based splitting of curves into linear segments used in [220]. A sample set of scale adaptive edge segments can be seen in Fig. 2.8(b). The higher the noise the shorter the edge segments are. In the worst case the chains are broken down all the way to edgels (edge pixels with orientations). The smaller the number of surrounding chains, the longer the resulting edge segments are. For example the neck of the giraffe in Fig. 2.8(a) is surrounded by more chains compared to that in (c). Hence the scale adaptive edge segments of neck of the giraffe in (b) are shorter than those of the giraffe in (d).

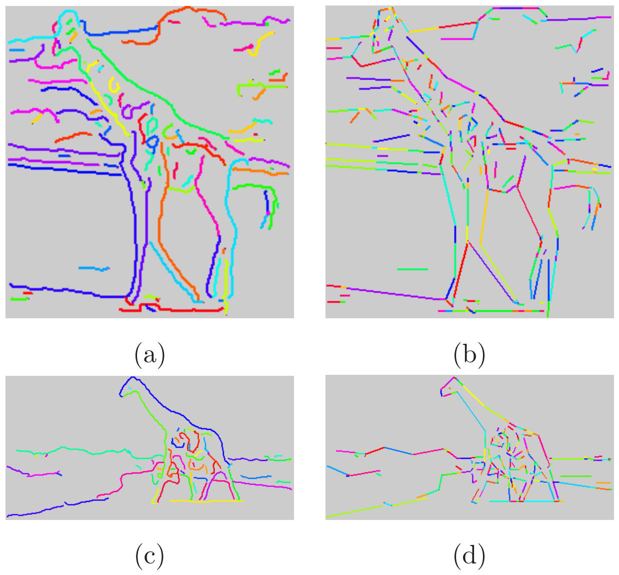


Figure 2.8: (a, c) Chains of edge pixels extracted using [149]. (b, d) The edge segments obtained by adaptively segmenting the chains. The chains and segments are shown in different random colors using the visualization tool of [149].

Scale adaptive edge segments are generated as follows: First, the chains are approximated by polylines using the publicly available code, [149]. Then each segment is split into smaller segments using the projections of the end points of neighboring segments. Only the end points within twice the range of the look ahead parameter, L (used in simulating prior (Section 2.4.2)) are used. Thus the number of pieces a segment is split into depends on the spatial distribution of segments around it. Fig. 2.9 demonstrates these steps.

To mitigate the effect of discretization on the turn angles at the junction points, we introduce virtual edge segments at those points. They are used to generate a set of SPs that can captures symmetries involving junctions in a better way. The orientation of a virtual segment at a junction is set equal to the mean of the orientations of the segments incident at the junction point. A sample set of virtual segments can be seen in Fig. 2.10.

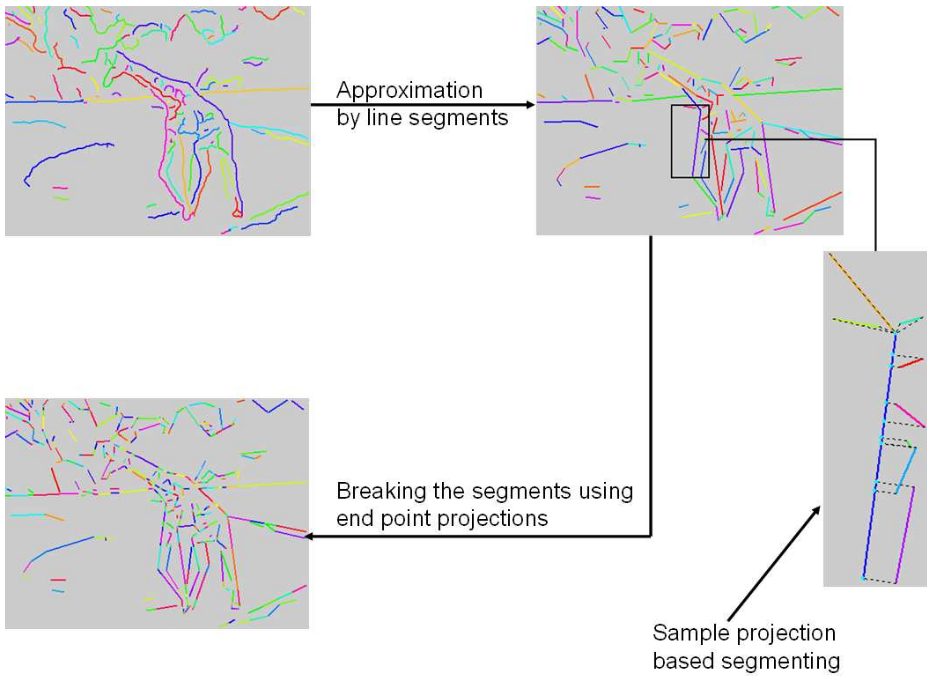


Figure 2.9: Process of splitting edge links based on spatial distribution of chains. The chains are approximated by line segments. Each segment is then broken by the projections of the end points of the neighboring segments. In the sample, the long blue segment is broken by the projections from its neighboring segments. The projections are shown as dotted lines and the break points in cyan.

Now we explain the state space of symmetric points (SPs) which captures local symmetries of our *scale adaptive edge segments*. Since the number of such edge segments is much smaller than the edge pixels, the number of SPs is significantly reduced in comparison to the full space of SPs generated by all edgel pairs. This reduced space of SPs forms the locally symmetric space for our virtual robot, and hence our robot marches faster, which improves the computation speed. For each pair of the edge segments we compute an SP as follows. We first sample equidistant points on each of the edge segment. Orientations of the points are set to the orientations of the respective edge segments. Then we compute the SPs between all pairs of the sampled points. Only the best SP according to the quality measure in Eq. (2.10) is selected as the SP induced by the two edge segments. The reduced and the full space of SPs are shown in Fig. 2.11. The reduced space is shown as red dots while the full space is shown as black dots.

Now we explain the values of the parameters used in our system. G_σ, S_σ, L are used in simulating the prior based proposal distribution, and D_σ, R_σ are used in computing the importance weights of the particles. We set $D_\sigma = 0.7, R_\sigma = 0.2$ since the readings from the model are normalized using scale factor. That is we are more tolerant to the deformability in the shape of the skeleton

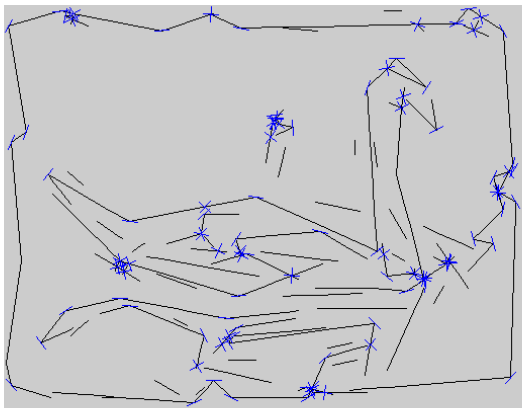


Figure 2.10: Virtual segments are introduced at the junction points to mitigate the effect of discretization in generation of symmetric points. The junction points are identified by intersections of the line segments within certain range. The orientation of a virtual segment at a junction is the average of the orientations of the segments inducing the junction point. The virtual segments are shown in blue. Their lengths are magnified for visualization.

but stricter with the deviation in the radii. The heuristic behind such choice is that under non-rigid transformations the shape of a skeleton path changes but the radii along the path do not. Hence we can be more tolerant to the skeleton paths. G_σ is set equal to the average length of the scale adaptive edge segments. The heuristic behind this is that the longer the scale adaptive segments the less noisy the image is and hence we can be more tolerant to gaps in the boundary. We set $S_\sigma = 45°$ in order to tolerate possible large inaccuracies in the directions of edges. L is set to be equal to $e_\mu + 0.5e_\sigma$ where e_μ is the average length of the edge segments approximating the edge chains before being split into scale adaptive segments. e_σ is the standard deviation of the lengths of the segments. The heuristic is that the longer the edge segments are the further the robot should look ahead to have meaningful followers.

2.7 Experimental results

We demonstrate the results of our technique using ETHZ dataset ([88]). We selected this data-set since it includes pre-computed edge images and model shapes for each class. The edge chains are obtained using [149]. For initialization a threshold on the gradient strength is applied. The robot is initialized around end points of all strong chains. An example initial set of poses for the robot can be seen in Fig. 2.12(a). After several iterations the robot localizes itself on to the object of interest because of model constraints. The process can be seen in the Fig. 2.12 (a)-(c). Also a video showing the process is attached as a supplementary material to this paper. This is similar to global localization in robot mapping domain where the initial position of the robot is

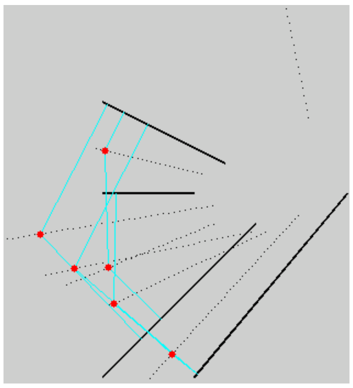


Figure 2.11: SPs (black dots) are computed between pairs of sampled points from different segments. The best SPs according to the quality measure based on the deviation angles are shown as red dots and the corresponding intending lines are shown in cyan.

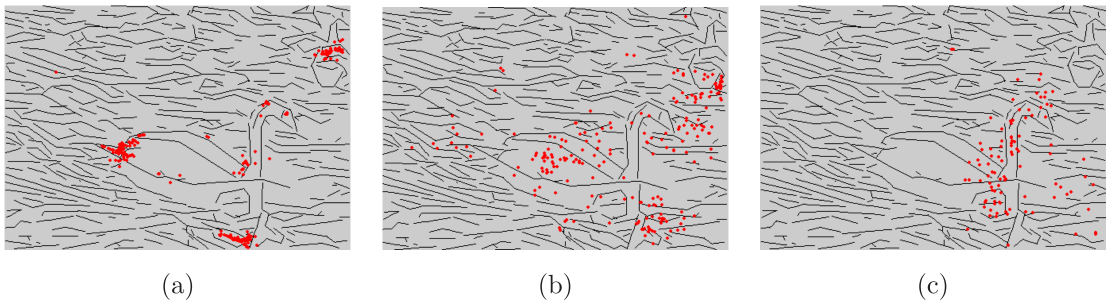
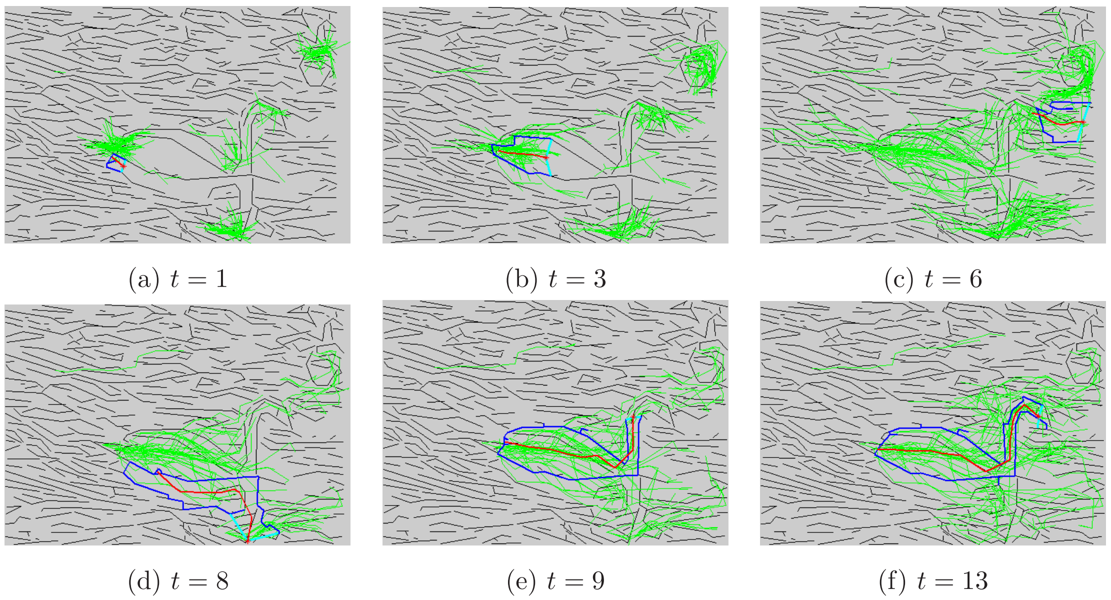


Figure 2.12: (a) The initial set of particles, $\{\mathbf{x}_1^{(i)}\}_{i=1}^{N_p}$ sampled around the end points of thresholded edge chains. (b) and (c) The distribution of $\mathbf{x}_t$s at $t = 5$ and 10. As the filter progresses the particles are concentrated around the object of interest, because of the model constraints.

unknown ([95]). The difference between localization and *global localization* is that in localization, the robot's initial pose is known while in global localization its initial pose is unknown. We would like to note that a threshold on gradient strength is used only for for initialization of particles not when the filtering is in progress. The filtering process can be viewed as multi-hypotheses tracking where several hypotheses for skeletons and contours are maintained to avoid getting stuck in local optima of the grouping process. A sample evolution of the filtering process can be seen in Fig. 2.13. The green curves are the multiple hypotheses for the skeletons while the most likely one is shown in red. The most likely boundary is shown in blue. A sample recovery from local optima can be seen from Fig. 2.13(c), (d) and (e).

The filtering stops when the MAP estimate of the posterior $p(\mathbf{x}_{1:t}, c_t|Z_m, U_g)$ drops below a certain threshold Δp. This stop criterion is needed so as to make partial grouping possible. Also it is


(a) $t = 1$ (b) $t = 3$ (c) $t = 6$

(d) $t = 8$ (e) $t = 9$ (f) $t = 13$

Figure 2.13: The evolution of the particle filter. The multiple hypotheses for the skeletons are shown in green. The most likely skeleton is shown in red and the most likely boundary is shown in blue. The space of SPs is shown as gray points.

important to note that this posterior is different from the prior probabilities, $p(x_t|x_{1:t-1}, U_g)$ of the follower symmetric points ($x_t s$). With enough particles this allows for both grouping "non-smooth" contours that confirm the model and distinguishing smooth contours that do not confirm the model. The best particles in all the modes of the posterior are selected as the skeletons, which induce the target contours. Typically the number of modes in the posterior is equal to the number of objects satisfying the model.

We present results on three classes of objects swans, bottles and giraffes whose model shapes are shown in Fig. 2.2. A sample of our results can be seen in Figs. 2.14, 2.15 and 2.16. Many more results are included in the supplementary material submitted with this paper. The large number of edges in the test images that do not belong to target contours demonstrates that we are able to group contour segments in the presence of distracter segments between locally symmetric contour pieces. Thus the proposed approach has strong potential for applications on real images.

The shape variability of grouped contours in Fig. 2.14, which were all obtain with a single shape model of a swan, demonstrates the flexibility of the proposed shape model. The second column of Fig. 2.15 shows an example of grouping multiple objects, while the fourth column shows an example of grouping partial contours when occluded by other objects. The generality of our symmetry set based model can be seen in Fig. 2.16, where we are able to ignore the shape details of legs of the giraffes and capture the significant parts of the giraffes namely the neck and the body.

It is important to note we are not able to find objects whose orientation is very different from the orientation of the model image. For example, in the third column of Fig. 2.14 we are able to group only one swan. Since we normalize the model readings to the size of the model shape we expect a scaling factor to be known for the object we are trying to group. The scaling factor is currently set manually.

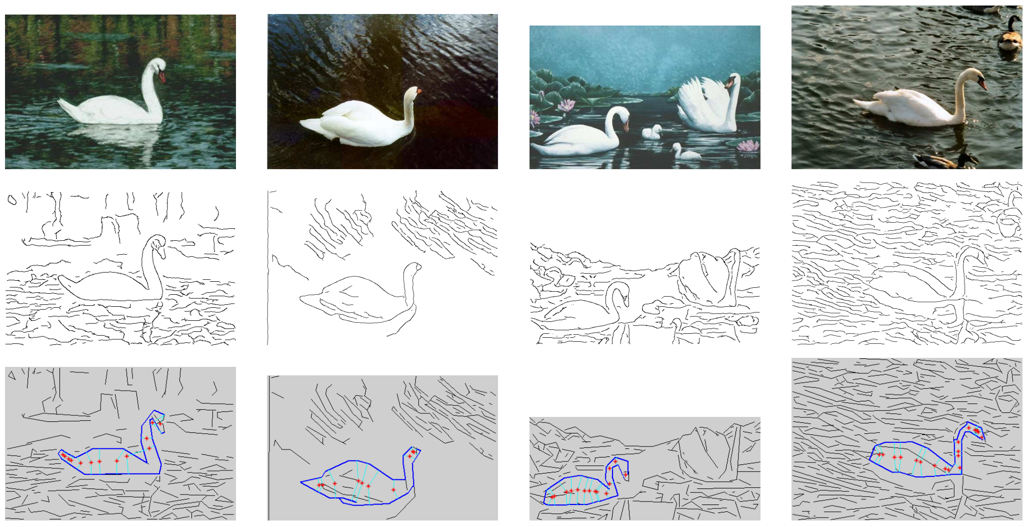


Figure 2.14: Grouping results on swan images. Top row original images, middle row edge images, and bottom row grouping results.

Our framework seems similar to shape-based object detection. But there is a key fundamental difference to the other object detection techniques (like [286, 86]): we build matching tokens and perform matching *simultaneously*. For e.g. our grouping tokens can be edgels while the other approaches *require* a preprocessing step of grouping contour pieces using low-level techniques. Hence our experimental set-up is quite different and is similar to that in shape-prior based segmentation in [159], for example, we run the swan images with swan model. Further the detection success is usually measured using bounding-box overlap. In our case success is measured by object boundary overlap which tends to be more precise detection. Therefore an exact and fair comparative evaluation of our system is arguably not possible. However we perform quantitative and comparative evaluation of our system as described below.

For each class we run our system with the class specific model. We report the retrieval rates based on the number of instances whose contours were grouped successfully. Even though a fundamentally different paradigm we ran the publicly available system of [86] in a similar fashion i.e. each class with the specific class model. The key and fundamental differences between our system and [86] are:

- We consider the object is successfully retrieved if we group at least 80% of the boundary of the

Class	Total instances	F-50%	F-60%	F-70%	F-80%	**Our method**
Swans	16	93.75%	93.75%	81.25%	62.5%	**87.5%**
Bottles	30	91.667%	87.5%	83.333%	54.84%	**80%**
Giraffes	44	82.222%	75%	59.09%	15.91%	**45.45%**

Table 2.1: Comparison between [86] and our method. F-x% denotes the retrieval result of [86] using x% of bounding-box overlap. See text for details.

Class	Total instances	% Retrieved
Swans	33	81.82%
Bottles	55	85.45%
Giraffes	91	51.65%

Table 2.2: Retrieval results using our method on the ETHZ dataset.

target object present in the edge image. The detection success in [86] is determined by a bounding box overlap percentage. We report the results for overlap percentages of 50%, 60%, 70% and 80%. The bounding box overlap of 80% appears to be most comparable to our definition of successful retrieval.

- In our system we use the *single* hand-drawn models (shown in Fig. 2.2) for each class. In [86] each class has multiple models learnt from a training subset of the instances. Therefore, we report only the retrieval rate of the best performing model for their system. This gives their system a clear advantage.

In Table 2.1 we report comparisons of the two methods on the same test images and test instances of target objects. We would also like to note that the bounding box accuracy does not necessarily imply accuracy in *boundary* detection. In [86] they use a subset of the dataset to learn the model and hence their test instances are only a subset of the dataset. Since we do not learn any model we can test our system on all instances. In Table 2.2 we present quantitative evaluation of our system on all instances for each class.

The number of particles needed for our experiments are determined empirically. On average we need about 200 to 500 particles. In general the number of particles required depends on the amount of clutter in an image. This is because particles can be viewed as multiple hypotheses needed to make delayed decisions after enough evidence has been accumulated.

The most computational burden of our system in simulating the prior distribution (described in Section 2.4.2). Prior has to be simulated for every unique particle. The average grouping time for noisy images is 5-8 minutes while for less noisy images it is about 2-4 minutes on a computer with Intel(R) Pentium(R) D CPU with 3.40 GHz and 0.99 GB of RAM. We would like to note that our system combines local information sequentially to make global optimal decisions so the speed of our system is quite independent of the resolution of the image unlike some methods where they build the global information of the image and then optimize it using graph-based techniques (for e.g. [241]).

2.8 Conclusion and discussions

Object detection and recognition by shape has a long history in computer vision. The basic idea is to extract some shape features and define distances between shapes using those features. Many heuristics are used in designing the features and distances so that the distances can be used to label or classify objects in accord with human perception. Some examples include [233, 18, 176, 14, 279]. Though such methods have some invariance capabilities to handle some deformations they cannot handle clutter and artifacts which commonly occur in real images. It is important to note that this is not their limitation per se because they are designed assuming the objects are already separated from the clutter or background. Separating objects from background has been a low-level vision problem for many years but for fully automatic segmentation it has to be combined with a recognition process. Hence researchers have been focussing on unified approaches for object recognition and segmentation ([29, 173, 159, 254, 266]). Unifying object detection/recognition and segmentation/grouping typically involves adding high-level prior about shapes of objects to the low-level discriminative approaches used for segmentation and/or grouping ([159, 264, 262]). It may also involve quantifying low-level tasks for improved detection/recognition ([255, 191, 265]).

Detecting objects can be at various precision levels starting from roughly detecting bounding boxes around the objects to finding the boundaries of the objects and even to finding the full interior contours of the objects. Object detection involving contours can provide a basis for better high-level applications which require more precision like medical imaging, human robot interactions etc. The first step in detecting object boundaries is extracting contour parts from images. Contour parts are then grouped to form object boundaries using methods that are essentially trying to perform shape matching in real images. For example, [230, 207] use Chamfer distance ([30]) to match fragments of contours learnt from training images to edge images, but their results are typically grassy contours. [86] uses a network of nearly straight contour fragments and sliding window search. More recently [286] formulated the shape matching of contours (identified using [285]) in clutter as a set-set matching problem. They present an approximate solution to the hard combinatorial problem by using a voting scheme ([266, 174]) and a relaxed context selection scheme using linear programming. They use shape context ([18]) as shape descriptor.

Our direction of research is to group more primitive tokens (very small linear segments) instead of parts (chains of pixels encoding some shape) using local symmetry based shape constraints *sequentially* to group object boundaries directly. The main reasoning underlying such approaches is that parts produced using low-level cues alone are prone to unrecoverable errors. It is similar to the reasoning behind incorporating shape prior in segmentation ([159]). We map the problem of contour grouping to a SLAM problem as it is stated in the field of robot mapping. We extend the particle filter based approach used in SLAM so that statistical inference based on a reference model is possible. In comparison to previous approaches our work has at least two serious advantages that are demonstrated in our experimental results. We are able to group contour segments in the presence of distracter segments between locally symmetric contour pieces. Even if we our shape models are derived from complete contours, grouping of only parts of contours is possible.

Contour grouping is a very hard problem that requires low-level, mid-level and high-level constraints. The proposed approach provides a versatile and robust framework for integrating all the three types of constraints. The shape constraints currently are based on single paths in symmetry sets that capture major parts of the object. For some complex objects shape constraints cannot be represented using single paths. Multiple paths in a symmetry set can be viewed as a collection of single paths in a "shape-tree". Different paths capture different parts of the shape. These parts can be connected using a tree structure. Our system can be extended to accommodate multiple paths in at least two different ways:

- Our system can be run for each path separately to identify all major parts and then part-matching can be performed using techniques like [286, 171]. This requires integration of tree-based shape matching systems on top of our system.
- Perform multi-robot mapping: different virtual robots can explore different single paths and can collaboratively combine the parts together. This requires new implementation for *joint* posterior for different paths and contour maps of parts.

The former approach can also be viewed as map merging problem studied in multi-robot mapping as in [43, 5] while the later approach as traditional collaborative mapping as in [129, 93]. We intend to develop a system that can handle multiple path based representation as part of our future work.

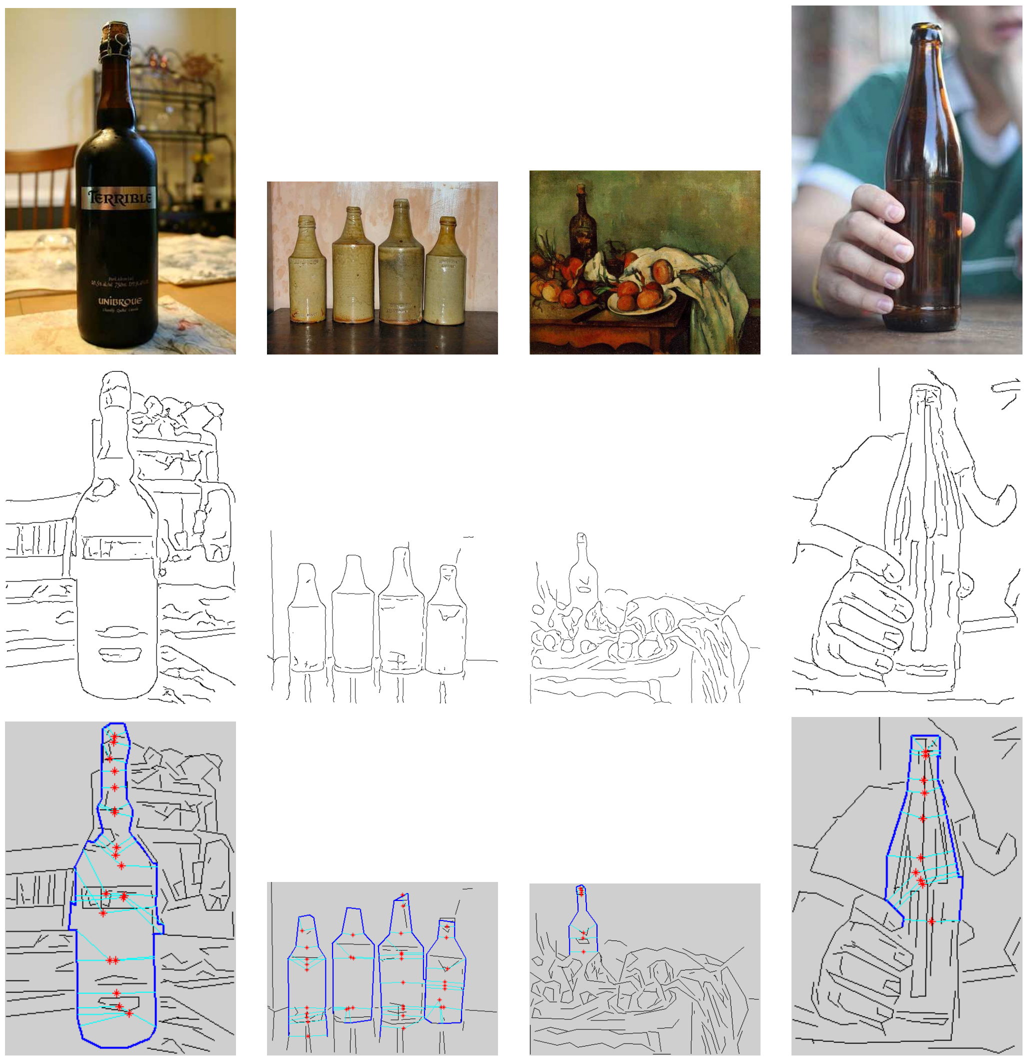


Figure 2.15: Grouping results on bottle images. Top row original images, middle row edge images, and bottom row grouping results.

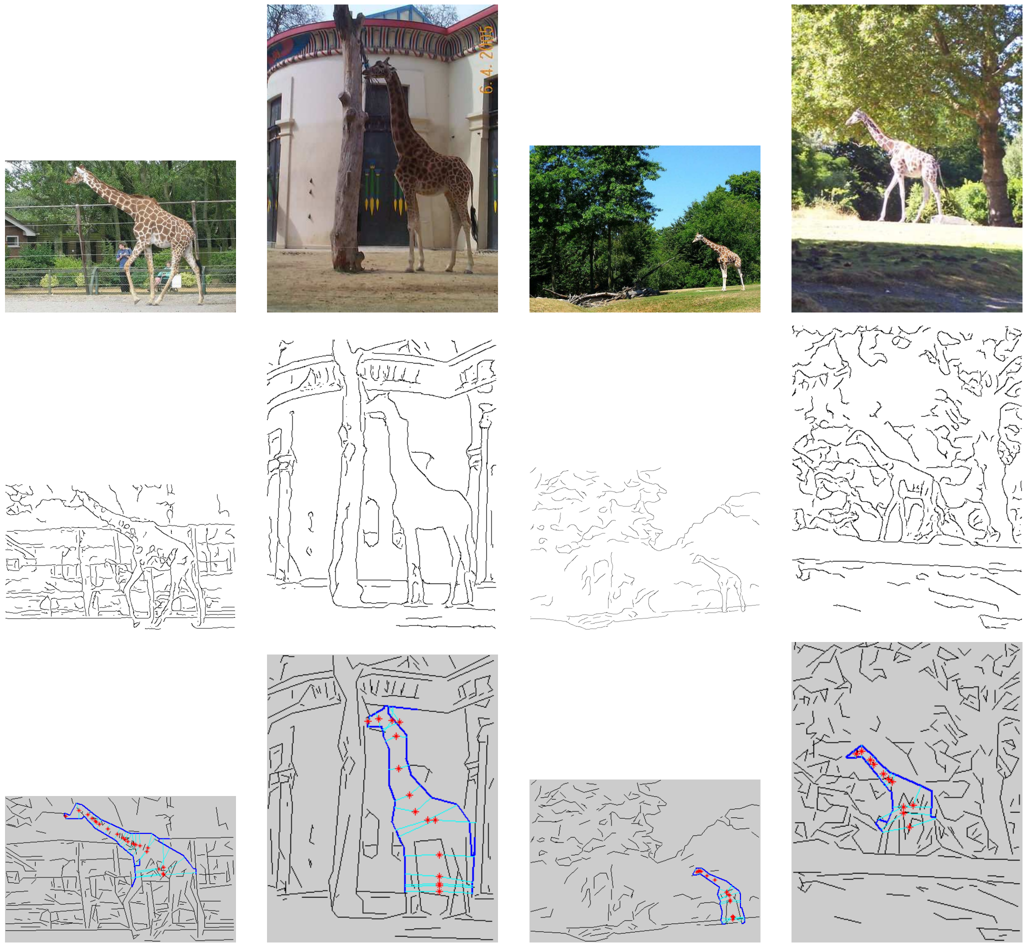


Figure 2.16: Grouping results on giraffe images. Top row original images, middle row edge images, and bottom row grouping results.

CHAPTER 3

Simultaneous Skeletonization and Contour Grouping

The key idea of this paper is to group contours by taking advantage of duality properties of contours and skeletons. Regularizing contours and skeletons *simultaneously* allows us to combine both low level perceptual grouping constraints as well as higher level radius constraints. This leads to robust extraction of occluding contours in real images in the presence of distracting contours, resulting from inner structures or noise. A sequential Monte Carlo estimation method is employed to maximize the joint posterior representing contours and skeletons. Our experimental results demonstrate the practical applicability of our approach on real-world edge images.

3.1 Introduction

Medial axes (MAs) and object boundaries, also known as skeletons and contours [27], are very useful descriptors for object recognition. According to Blum's definition a skeleton S of a set of object boundaries D is the locus of the centers of maximal disks. A maximal disk in D is a closed disk contained in D that is interiorly tangent to the boundary of D and that is not contained in any other disk in D. Each maximal disc must be tangent to the boundary in at least two different points. A set of skeleton points $s \in S$ and the radii $r(s)$ of their maximal disks can be used for reconstructing the boundary without any ambiguity. An important property is that the skeleton can be computed for every planar shape, but computation of skeletons without boundary information is not possible. Extracting occluding contours using only edge-detection algorithms without any regularization is an ill-posed problem due to ambiguities in the gradient space of real-world images.

In this paper we present a new sequential and *simultaneous* estimation of medial axes and the contours. The main idea is to cast the contour grouping problem as SLAM (Simultaneous Localization and Mapping) of a virtual robot. This idea was first introduced in [3]. In this approach, the skeleton is treated as trajectory of a virtual robot and the boundary is formed as a map of associated edge segments. A joint-posterior on the trajectory and map is then maximized using Rao-Blackwellized particle filtering with constraints from a reference model. Thus in effect they regularized the contour growth by using a shape reference model represented a set of skeletal paths which *naturally* enforced local symmetry. The main difference of the presented approach to that of [3] is the fact that we do not use any *explicit shape models* of objects to be extracted. Since we integrate grouping constraints based on contour-skeleton duality [50] into the SLAM framework we are able to use radius constraints on the medial axes points that are based on more generic geometric properties than shape. The underlying assumption for our framework is that occluding contours form *prominent* structures and their skeletons are well-behaved in the sense of satisfying the radius constraints. A detailed analysis of medial axis properties and algorithms is given in the recent book by Siddiqi and Pizer [235].

We conclude the introduction with an algorithmic overview of the proposed approach as shown in Fig. 3.1. For a given input image, edgels (edge pixels with orientations) are first extracted. The edge detection is performed using the *pb* detectors [188]. Since the ability to obtain edge orientations plays an important role in our approach, we use a very recent third order edge detection approach [244] to compute the directions of the edge pixels detected using *pb*. Third order gradient information yields significantly more accurate orientations of edge pixels than those by previously known methods.

Then we compute center points (CPs), which are candidate MA points following the geometric construction in [3]. Since our system does not use any shape models, the initialization of the particles or in other words boostrapping the filter, becomes an important problem. Section 3.3 presents our EM based solution to the problem. Then the posterior representing boundary and the skeleton is maximized using particle filtering as described in sections 3.4 and 3.5.

Our experimental evaluation in section 3.6 clearly demonstrates that the proposed approach outperforms recent state-of-the-art algorithms in obtaining occluding contours.

3.2 Related work

Contour grouping remains to be an important open problem and is a very active research field as documented by the recent papers [285, 253, 244, 124, 242, 100].

Different cues like closure, good-continuity [194], and more recently motion [242] and symmetry [241, 179] have been used to find contours. We exploit the symmetry cue since symmetry is known to play a crucial role in object recognition both by humans and computers [181, 214, 175, 287, 142]. Thus obtaining contours based on symmetry cues are expected to be more useful.

Symmetry for rigid and non-rigid objects can be captured in several ways like for example using ribbons [194], using planar reflections [212] or more classically using medial axes [27]. While different representations of symmetry have different advantages based on the context of application,

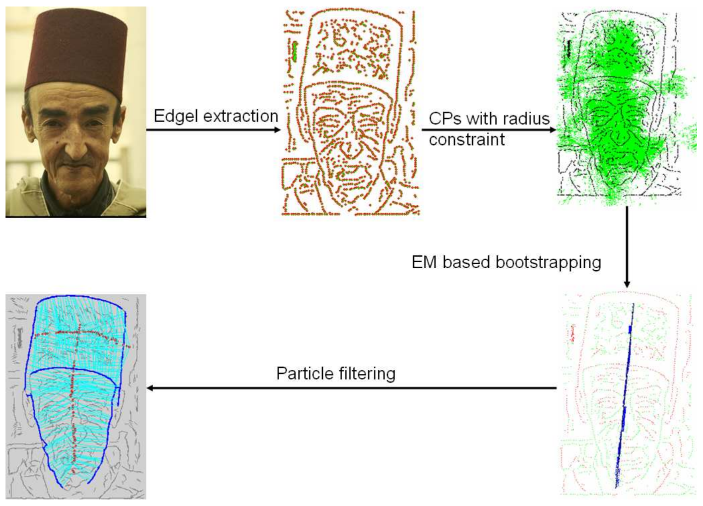


Figure 3.1: Overview of our algorithm. (1) The edgels (edge pixels with orientations, gradient strength is not used) are extracted using a combination of [188] and [244]. (2) Then, edgel pairs are used to construct center points and radius constraints are used to find CPs in the meaningful regions of the image. (3) Using the constrained set of CPs, EM is employed to find most likely skeleton curves approximated by line segments in $3D$ space, the position (x, y) and radius (r) of the CPs, [170]. The CPs around the segments are used to bootstrap the filter. (4) Grouping and radius constraints are used to maximize the posterior representing skeletons and boundary using particle filtering approach.

those that can capture local symmetry play more important role in grouping contours.

Probabilistic reasoning is popular in processing noisy images. Examples include [133] that is used for tracking contours in cluttered background and [25] that is used for detecting and tracking of motion boundaries. Also the popular set of edge detectors, *pb*, is built using probabilistic representation of boundary [188]. Our approach represents a joint-posterior of *both* contour and the skeleton. Since particle filtering provides a strong framework with Bayesian reasoning to capture complex (non-linear and non-gaussian) dynamics of probability density functions, we use that approach.

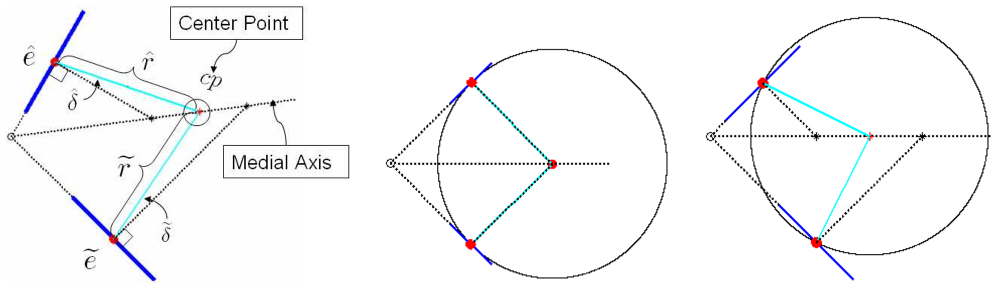
A particle filter algorithm called JetStream was applied to contour detection in [209]. The detection is performed on edge pixels with particles following the contour directly. In this paper a simple ribbon geometry is also used for road extraction. This approach is a static version of a particle filter tracking algorithms, where temporal sequence is replaced with a sequence of contour points. JetStream often fails to track contours in complex images, since only low level features (image gradient and corner detection) are used as the basis for particle observations. Therefore, it requires user interaction during the tracking process.

[3] first cast contour grouping as SLAM problem. Since particle filtering is a successful approach for SLAM they used particle filtering for contour grouping as well. Our approach extends the idea and is also related to the recent work on *exploration* and active SLAM [47, 239] where the robot chooses its actions by distinguishing between explored and unexplored regions. Since we use a strong statistical framework with constraints that are not entirely low-level, our work has advantage of capturing "relevant contours" even in the presence of distractions. While the techniques like [209] use good statistical framework they do not exploit geometric information. The methods like [179, 219] exploit geometry but are limited in noisy conditions.

3.3 Center point space and EM based bootstrapping

We first describe the space of center points (CPs) for a given input image. As introduced in [3], each pair of edgels determines one CP as the center of the circle which is tangential to both the edgels.The maximal property of the disks cannot be determined at this stage of our computation as the true object boundaries are unknown. Hence the set of CPs forms a superset of true MA points.

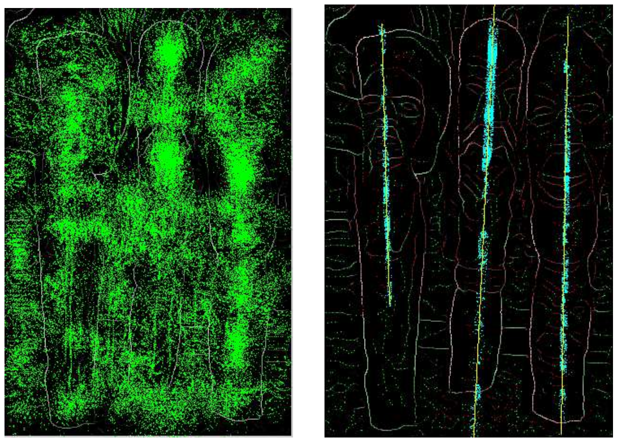
Now we are ready to introduce the state space using which skeletons and boundaries are estimated, see Fig. 3.2 (a): each state (or pose) $\mathbf{x}_t$ is a vector composed of the coordinates of a CP (cp), coordinates of its two associated edgels ($\widehat{e}, \widetilde{e}$), and their directions. The radius of the center point, r, is the average of the "intending radii" $\widehat{r}$ and $\widetilde{r}$. Because of the discretization and noise in edge orientations we allow the disc to be *approximately* tangential. For example, the centers of both circles in Fig. 3.2 (b),(c) show two CPs whose associated pairs of edgels is shown with red dots and their directions with blue lines. Both CPs are acceptable in our approach and hence the set of CPs forms a superset of even the symmetry set [105]. The quality of the center point depends on the average of the "deviation angles" $\widehat{\delta}$ and $\widetilde{\delta}$ which determines the tangentiality of the disc.



(a) (b) (c)

Figure 3.2: (a) Components of a center point. (b) Perfect center point where inscribing circle touches the edgel pairs tangentially. (c) Imperfect center point where the inscribing circle goes over the edgels.

We will now consider the problem of initializing the particles. A set of CPs whose radii belong to a range of radius ($\Re$) that is expected to capture the symmetric parts of the prominent structures in the image is used to find highly likely skeleton curves. For this step we assume that a skeleton curve can be approximated by piece-wise linear parts and fit line segments in the $3D$ space i.e. the position, (x, y), and radius, r of CPs. We use the extended Expectation Maximization (EM) algorithm introduced in [170] for line fitting. [170] finds several line segments but only those that have high degree of mirror symmetry in the distribution of the CPs are selected as the final set. Now the CPs around the line segments are used as the initial set of particles i.e. $\{\mathbf{x}_1^{(i)}\}_{i=1}^{N_p}$. Fig. 3.3 shows the filtered set of CPs and mirror symmetric line segments for the "pillars" image. For the current system the ranges of radius are manually set but they can also be learnt using a set of training images. Also some images might have a *set* of radius constraints depending on the variance of the sizes of its prominent parts.



(a) (b)

Figure 3.3: (a) A set of center points with radii in the range $\approx [50, 75]$ pixels wide. (b) The mirror symmetric line segments generated using [170]. CPs around the line segments (shown in cyan) are used for bootstrapping the filter.

3.4 Maximizing the joint posterior of contours and skeletons

In this section we explain how the joint posterior representing the contours and skeletons is maximized. The underlying idea is that contour grouping problem can be cast as a mapping problem of a virtual robot which walks in the state space induced by the CPs. We first briefly go over a standard solution to the mapping problem and then describe how we adapt that solution for the grouping problem.

3.4.1 Mapping

The mapping problem is to maximize the joint posterior of robot trajectory, $x_{1:t}$, and the map of the environment, m_t, it is walking in. A sequence of range measurements, $z_{1:t}$, and odometry measurements, $u_{t:t}$ are used as constraints. Thus the goal is to find $\underset{x_{1:t},m_t}{\operatorname{argmax}}\, p(x_{1:t}, m_t|z_{1:t}, u_{1:t})$. Since both the trajectory and map are estimated simultaneously the problem is called *Simultaneous Localization and Mapping (SLAM).* To allow for online optimization, *sequential* Monte Carlo estimation called *particle filters* have been successfully applied [251]. The complexity of the particle filters is usually measured in the number of particles (random samples) needed to represent posterior of the state. The dimensionality of the state being estimated plays a crucial role in efficient application of particle filters. Rao-Blackwellization allows to reduce the dimensionality by factorizing the states that can conditionally and analytically be estimated. Since a map m_t can be analytically constructed for a given sequence of poses $x_{1:t}$ and observations $z_{1:t}$ ([200]), Rao-Blackwellization of the joint posterior results in [251]:

$$p(x_{1:t}, m_t|z_{1:t}, u_{1:t}) = p(m_t|x_{1:t}, z_{1:t})p(x_{1:t}|z_{1:t}, u_{1:t}) \tag{3.1}$$

Once the state is decomposed any standard particle filter can be used, the most popular being Sampling Importance Resampling (SIR) filter [117]. Each particle $x^{(i)}_{1:t}$ is a candidate sequence of poses and has an associated map $m^{(i)}_t$ constructed analytically upto time t. At every time step t, the robot uses the most likely particle, i.e. the i^{th} particle that maximizes $p(x^{(i)}_{1:t}, m^{(i)}_t|z_{1:t}, u_{1:t})$ for localization and navigation.

3.4.2 Grouping

Similar to the above problem the goal here is to maximize the joint posterior of the trajectory (skeleton), $\mathbf{x}_{1:t}$ of a virtual robot and the map (contour), c_t, of its environment (image). It is important to note that the trajectory of a robot in case of SLAM is a sequence of poses, i.e. its positions and heading directions while in the case of grouping it is a sequence of center points as described in section 3.3. In [3] the constraints were obtained from an explicit shape reference model as a sequence of expected radii, $r_{1:t}$ of the maximal discs sampled along a skeletal path in the model image. Since c_t can be analytically computed given a sequence of center points $\mathbf{x}_{1:t}$, [27] the state of the joint posterior can be decomposed using Rao-Blackwellization as:

$$p(\mathbf{x}_{1:t}, c_t|r_{1:t}) = p(c_t|\mathbf{x}_{1:t})p(\mathbf{x}_{1:t}|r_{1:t}) \tag{3.2}$$

Following the approach in [117], [3] used SIR particle filter with a complex optimal proposal according to Doucet et. al. [63]. At every time step t, the contour $c^{(i)}_t$ associated with the most likely particle that maximizes $p(\mathbf{x}^{(i)}_{1:t}, c^{(i)}_t|r_{1:t})$ is declared the "part of the contour grouped". Once the robot uses up all the sequence of radii from the model, the filtering stops. The major challenge in generalizing their approach is handling the shape deformability.

In this paper, we partially address that issue using more generic radius constraints. A range of radii of the CPs is used to capture geometric properties specific enough to be able to distinguish "objects of interest" from noise and generic enough to eliminate "explicit shape models". Thus our approach is neither entirely low-level (which has serious limitations) nor very high-level (which has serious challenges). The sequence of model radii $r_{1:t}$ is replaced with a set of grouping constraints Z that are based on contour-skeleton duality, good continuation and radius prior. Rao-Blackwellization in our case gives the following equation:

$$p(\mathbf{x}_{1:t}, c_t|Z) = p(c_t|\mathbf{x}_{1:t})p(\mathbf{x}_{1:t}|Z) \tag{3.3}$$

Since we use a *set* instead of a *sequence* as constraints a different proposal and importance weighting are employed. Hence we explain the first two basic steps of the SIR filter in our case, the remaining two steps of resampling and contour estimating are similar to that in [3]:

1) **Sampling/Proposal:** The next generation of particles $\{\mathbf{x}_{1:t}^{(i)}\}$ is obtained from the current generation $\{\mathbf{x}_{1:t-1}^{(i)}\}$ by sampling from a proposal distribution $\pi(\mathbf{x}_{1:t}|Z)$ which is assumed to satisfy the following recursion based on 1^{st} order Markov assumption:

$$\pi(\mathbf{x}_{1:t}|Z) = \pi(\mathbf{x}_t|\mathbf{x}_{1:t-1}, Z)\pi(\mathbf{x}_{1:t-1}|Z) \tag{3.4}$$

Therefore, each particle is extended $\mathbf{x}_{1:t}^{(i)} =< \mathbf{x}_t, \mathbf{x}_{1:t-1}^{(i)} >$ with $\mathbf{x}_t \sim \pi(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, Z)$.

2) **Importance weighting/Evaluation:** The importance weights $w(\mathbf{x}_{1:t}^{(i)})$ can be recursively estimated as:

$$w(\mathbf{x}_{1:t}^{(i)}) \propto w(\mathbf{x}_{1:t-1}^{(i)})\frac{p(Z|\mathbf{x}_{1:t}^{(i)})p(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)})}{\pi(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, Z)} \tag{3.5}$$

Since we do not use any shape model as a sequence of constraints we use $p(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)})$ as our proposal instead of the complex optimal proposal used in [3]. By substituting this proposal for $\pi(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)}, Z)$ in equation (3.5), we obtain

$$w(\mathbf{x}_{1:t}^{(i)}) \propto w(\mathbf{x}_{1:t-1}^{(i)})p(Z|\mathbf{x}_{1:t-1}^{(i)}, \mathbf{x}_t) \tag{3.6}$$

The details of the proposal, $p(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)})$, and particle evaluation, $p(Z|\mathbf{x}_{1:t-1}^{(i)}, \mathbf{x}_t)$, are explained in section 3.5.

Usually, the posterior we are tracking (i.e. the skeleton, $\mathbf{x}_{1:t}$, and the boundary, c_t) is multi-modal because of distractor segments and interior structures. It is important to have particles representing all meaningful regions of the posterior to be able to recover from distractions by noise. We use "prior boosting" ([108]) so as to capture multi-modal likelihood regions. In prior boosting more than one follower is sampled for each particle so that different followers can capture different modes of the likelihood of the posterior. The number of followers typically depends on the local geometric conditions: if there is lot of noise then lot of followers are automatically chosen and if there are less distractor segments then fewer followers are chosen. Thus after each iteration of Sampling

Importance Resampling we might have more number of particles $N_p' > N_p$ but we retain only N_p *with their weights*. An important difference between resampling a fixed number of particles from large number of particles is that we retain their weights while after the adaptive resampling step the weights are all set to be equal.

In contrast to stopping after the observations have been used up as in [3], the filtering process in our case is stopped when the likelihood of the posterior, $p(x_{1:t}, c_t|Z)$, of the best particle drops below a threshold, Δp. The trajectory and map of the best particle is declared as the skeleton and boundary respectively. It is important to observe that since our system groups boundaries in parts the skeletons obtained actually form a symmetry set which is a superset of the medial axis.

3.5 Proposal and evaluation

In this section we explain how the proposal $p(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(i)})$ and the likelihood $p(Z|\mathbf{x}_{1:t-1}^{(i)}, \mathbf{x}_t)$ are modeled. The former distribution is used to generate next generation of particles, $\{\mathbf{x}_t^{(i)}\}$ and the later is used to evaluate the new particles.

3.5.1 Local geometry ($\mathbf{x}_{t-1}^{(i)}$) in proposal with prior boosting

Assuming 1^{st} order Markov assumption the proposal is equivalent to $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$. Using $\mathbf{x}_{t-1}^{(i)}$ we estimate the contour parts added at $t-1$ and use two uniform circular neighborhoods as look ahead regions for possible extensions to the contour parts into "unexplored regions" (regions that are not enclosed by boundary pixels). The process of obtaining possible pairs of edgels can be seen in Fig. 3.4 (a). Thus at each step the virtual robot tries to extend the skeleton by expanding the contour. The number of followers depends on the number of edgels in the look ahead regions and the number of pairs that induce CPs within the radius range. All the center points induced by those pairs are equally likely according to $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$. The number of followers in our prior boosting is tightly upper bounded by $\Theta(mn)$ where there are m and n edgels in the two circular neighborhoods. If the noise is locally very distracting that is non-true edgel pairs inducing valid CPs, the number of followers is high. Since noisy edgel pairs are not expected to satisfy radii constraints for long time the weights of the particles with noisy CPs decrease as the filtering progresses.

An important parameter in the proposal is the size of the circular neighborhood which decides how far the virtual robot can "look ahead" to capture the likelihood regions of the posterior. If the robot looks too far ahead then it will be less sure if the boundary and skeleton extensions are true. On the other hand if it does not look far ahead enough it might be distracted by noise and stop exploring in the first place. An appropriate choice for the parameter is to use $\mathcal{C}_\sigma$ (introduced in section 3.5.2) which is the parameter that is used to measure good continuation in the evaluation of particles, since the followers beyond $\mathcal{C}_\sigma$ will be evaluated to have lower weights.

Before evaluating the weights for all the particles we cull several bad followers in similar

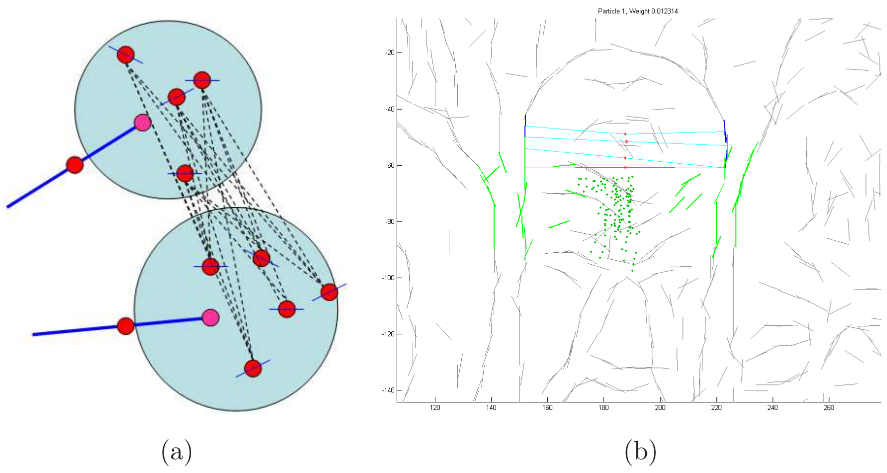


Figure 3.4: (a) Each edgel added to the boundary by $\mathbf{x}_{t-1}^{(i)}$ is extended according to its orientation (blue segments). The edgels in two circular neighborhoods are chosen as possible boundary extensions. All possible pairs of edgels that are used to induce the follower center points are shown in dotted lines. (b) The green points indicate the possible follower poses based on our proposal. These points are induced by the green edgels.

spirit to DP-SLAM [70]. We cull the followers which do not help in additional grouping of the boundary, those that violate the radius constraint by wide margin and those that extend boundary or skeleton with self-intersections. Fig. 3.4 (b) shows an example of follower center points (after culling) for a particular particle.

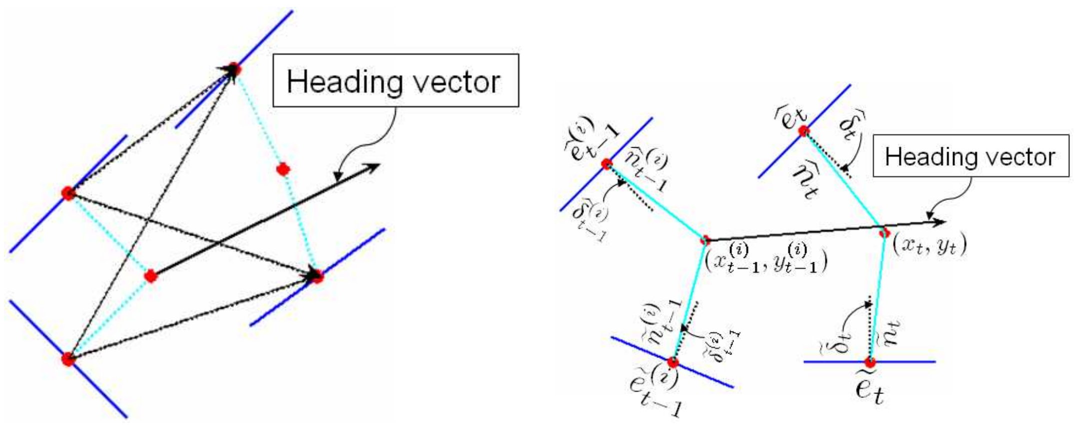
3.5.2 Evaluation based on the set of constraints (Z)

The particles are evaluated according to Eq. (3.6). Assuming the constraints are applicable only at the last step, $p(Z|\mathbf{x}_{1:t-1}^{(i)}, \mathbf{x}_t)$ is equivalent to $p(Z|\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t)$. It indicates the likelihood of $\mathbf{x}_t$ satisfying the constraints Z given $\mathbf{x}_{t-1}^{(i)}$. As mentioned in section 3.4, Z is a set of constraints obtained from the contour-skeleton duality ($\mathcal{D}_\sigma$), good-continuation ($\mathcal{C}_\sigma$) and the support for radius prior ($\mathcal{R}_\sigma$). The contour-skeleton duality constraint prefers the robot's paths along good center points. Fig. 3.2 (b), (c) show examples of a good and an imperfect center point. It is not possible for the robot to always be able to walk along perfect center points because of noise in the boundary directions and gaps in the boundary. The gaps are present because of discretization and lack of gradient information. $\mathcal{D}_\sigma$ is the parameter to be tuned in this regard which constrains the average deviation angles, $\widehat{\delta}_t, \widetilde{\delta}_t$ of the follower cp, $\mathbf{x}_t$ from getting worse. The higher the number of breaks in edge chains, the larger the $\mathcal{D}_\sigma$ has to be. The good continuation constraint prefers "smooth" trajectories and boundaries. This means branchy skeletons and boundaries violating Gestalt principles are penalized [269]. $\mathcal{C}_\sigma$ is the parameter to be tuned for such penalty which constrains the deviations in terms of pixels or sub-pixels.

The constraints from radius prior give us main advantage over pure low-level grouping techniques. The parameter, $\mathcal{R}_\sigma$, constrains the deviations of the poses from the radius constraints ($\Re$). The equation below shows the evaluation components:

$$p(Z|\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t) = d(\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t, \mathcal{D}_\sigma, \mathcal{C}_\sigma).s(\mathbf{x}_t, \mathcal{R}_\sigma) \tag{3.7}$$

The two components, d and s of the weight are explained using Fig. 3.5. The first step in computing



(a) (b)

Figure 3.5: (a) The heading vector (thick arrow) is the sum of the four vectors (thin arrows) connecting the two edgel pairs. (b) Components used in evaluation of a particle based on grouping constraints, Z.

the weights is to get the correspondences between the edgels at $\mathbf{x}_{t-1}^{(i)}$ and those at $\mathbf{x}_t$. We use the "heading vector" of the virtual robot to get the correspondences. The heading vector is computed as shown in Fig. 3.5 (a). The correspondences among the new and previous edgels are computed according to clockwise or anti-clockwise sweep ordering of the heading vector. In Fig. 3.5 (b) $\{\hat{e}_{t-1}^{(i)}$, $\hat{e}_t\}$ correspond to each other and $\{\widetilde{e}_{t-1}^{(i)}$, $\widetilde{e}_t\}$ correspond to each other. Once the correspondences are obtained d and s are computed using eqs. (3.8) and (3.9). In the derivations below d uses $\mathcal{D}_\sigma, \mathcal{C}_\sigma$

while s uses $\mathcal{R}_\sigma$.

$$\begin{aligned}
&d(\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t, Z) = \mathcal{H}(\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t, Z) + \mathcal{T}(\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t, Z), \text{ where} \qquad (3.8)\\
&\mathcal{H}(\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t, Z) = \mathcal{N}(\hat{h}, 0, \mathcal{D}_\sigma).\mathcal{N}(\|\hat{e}_{t-1}^{(i)} - \hat{e}_t\|, 0, \mathcal{C}_\sigma)\\
&\mathcal{T}(\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t, Z) = \mathcal{N}(\widetilde{t}, 0, \mathcal{D}_\sigma).\mathcal{N}(\|\widetilde{e}_{t-1}^{(i)} - \widetilde{e}_t\|, 0, \mathcal{C}_\sigma), \text{ where}\\
&\hat{h} = |\angle(\hat{e}_{t-1}^{(i)}, \hat{e}_t) - \angle(\hat{n}_{t-1}^{(i)}, \hat{n}_t)| - \min(\hat{\delta}_t - \hat{\delta}_{t-1}^{(i)}, 0)\\
&\widetilde{t} = |\angle(\widetilde{e}_{t-1}^{(i)}, \widetilde{e}_t) - \angle(\widetilde{n}_{t-1}^{(i)}, \widetilde{n}_t)| - \min(\widetilde{\delta}_t - \widetilde{\delta}_{t-1}^{(i)}, 0)
\end{aligned}$$

$$\begin{aligned}
&s(\mathbf{x}_t, Z) = \mathcal{N}(\Delta r_t, 0, \mathcal{R}_\sigma), \text{ where} \qquad (3.9)\\
&\Delta r_t \text{ is deviation of the radius of the maximal disc of } \mathbf{x}_t \text{, from the range } \Re,\\
&\mathcal{N}(x, \mu, \sigma) \text{ is the Gaussian (mean } \mu \text{ and standard deviation } \sigma\text{), evaluated at } x. \qquad (3.10)
\end{aligned}$$

3.6 Experimental results

In this section we present experimental evaluation of the proposed approach. Since we group parts of contours and skeletons the objects are grouped as a set of locally symmetric parts. For each image after extracting the edgels, radius constraints (ranges of radii of CPs) are chosen so as to capture the prominent parts of the "objects of interest". Then the virtual robot is initialized around the several segments produced by EM as described in section 3.3. For e.g. in the "pillars" image the robot is initialized along the segments shown in Fig. 3.3. After the filtering process stops (i.e. after the likelihood of the posterior drops below a threshold, Δp, see section 3.4 for more details), we are left with several best trajectories (skeletons, $\mathbf{x}_{1:t}$) and maps (contours, c_t) around each initialization. Each such symmetric part has a likelihood value, $p(\mathbf{x}_{1:t}, c_t|Z)$ associated with it, see section 3.4. All symmetric parts with the likelihood value below a certain threshold $p_{\min}$ are excluded. It is important to note that the likelihood values are not independent of t, for e.g. the "short-lived" explorations i.e. where t is too short will have small likelihood values which eliminates lot of unwanted groupings which for a short time satisfy the radius and Gestalt constraints. To summarize the parameters needed by our evaluation system are $\mathcal{D}_\sigma, \mathcal{C}_\sigma, \mathcal{R}_\sigma, \Delta p, p_{\min}$ which are all fixed once for a dataset and $\Re$, the set of radius constraints which are set per image.

Since our goal is to obtain occluding contours and there is no ground truth data yet available for just occluded contours in the Berkeley dataset, we perform a *qualitative* comparison of our approach with two recent approaches in [241] and [285]. Both techniques are mainly low-level but are based on geometric properties of the edge map. While, [285] extracts continuous edge-links by untangling cycles in graphs formulated using computational topology, [241] exploits *global* symmetry. Fig. 3.6 compares our result to [241]. The grouped contours by our method are shown in blue, and the corresponding MAs in red. While on an easy image like the laptop the results are comparable, our approach can capture all locally symmetric parts of the main object in the "stop-sign" image. Fig. 3.7 compares our result to [285] on some typical images from the standard Berkeley segmentation dataset [188]. Our method produces more information than just edge-links which can be better used for further processing like object recognition. For example, in the result from [285], the contours of the elephants are grouped, but the difference between the two elephants is not clear. In particular, both share a common contour curve. Our method clearly and correctly separates the parts of the two different objects from one another. Another interesting observation is that though [285] does not use gradient strength in grouping, their edge-maps look less cluttered than ours since we use a very low threshold to capture all gradients. This shows robustness of our approach. Integrating automatic learning of $\Re$ is part of our future work.

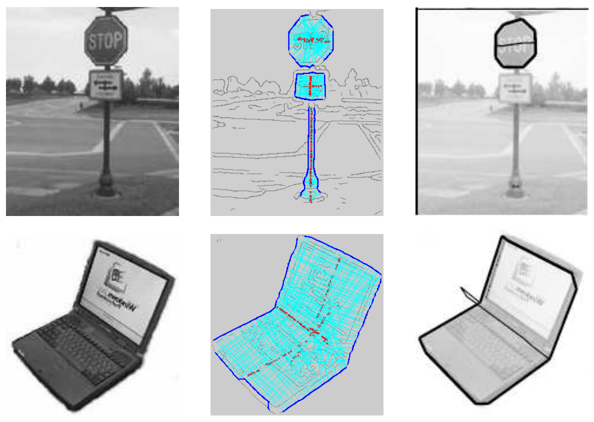


Figure 3.6: A visual comparison of our method (middle column) to the grouping results in [241] (right column). Because we do not rely on global symmetry we are able to capture different locally symmetric parts of the "stop-sign" image.

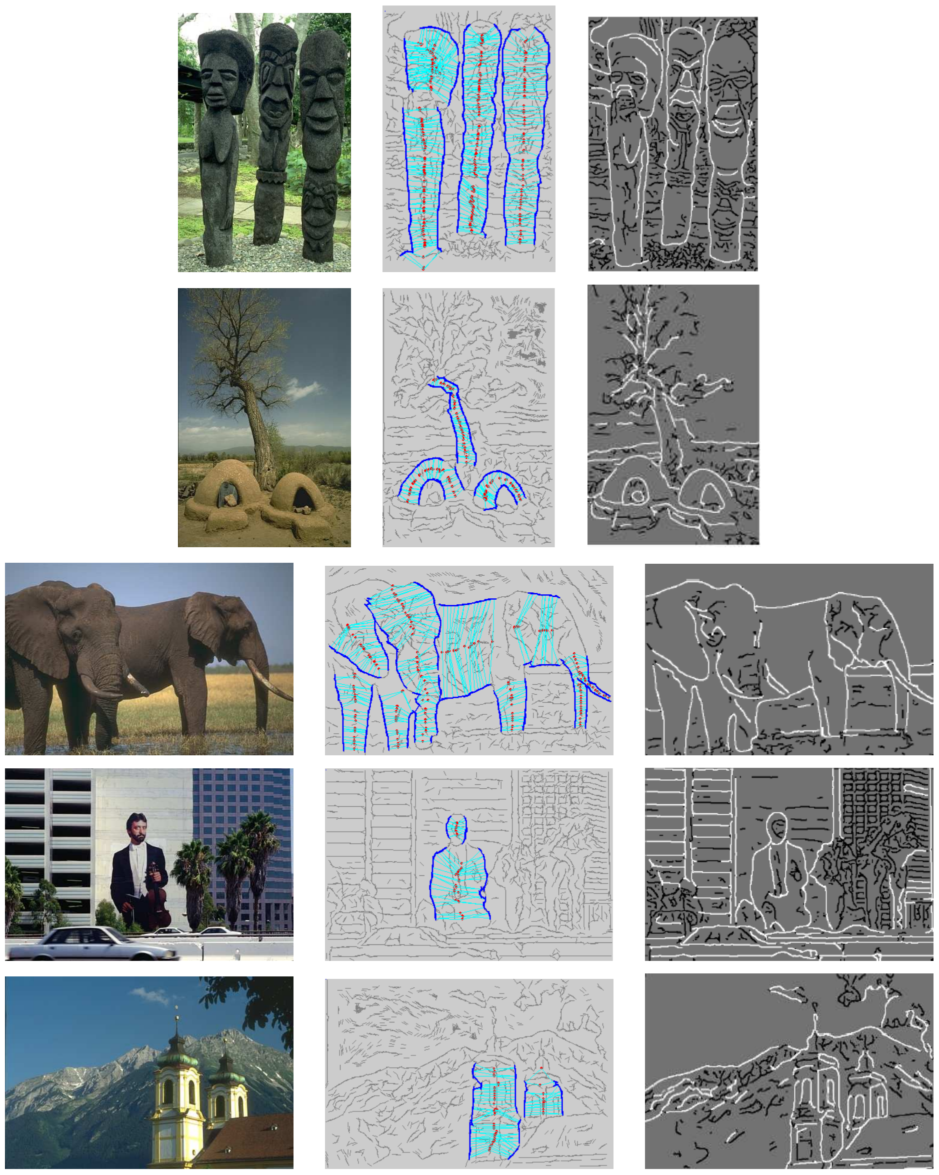


Figure 3.7: A visual comparison of our method (middle column) to the grouping results in [285] (right column). As can be seen even though our edge maps are noiser than those used in [285] we are able to group relevant *occluding* contours. This is possible because our approach relies on mid-level radius constraints. The advantage of using contour-skeleton duality constraints can also be seen in that we are able to capture the right side of the face of right-most pillar, which with pure contour constraints is hard to group without capturing other noisy links.

CHAPTER 4

Symmetry of Shapes via Self-Similarity

We describe a simple and novel approach to identify main similarity axes by maximizing self-similarity of object contour parts divided by the axes. For a symmetric or approximately symmetric shape, the main self-similarity axis coincides with the main axis of symmetry. However, the concept of the main self-similarity axis is more general, and significantly easier to compute. By identifying critical points on the contour self-similarity computation can be expressed as a discrete problem of finding two subsets of the critical points such that the two contour parts determined by the subsets are maximally similar. In other words, for each shape, we compute its division into two parts so that the parts are maximally similar. Our experimental results yield correctly placed maximal symmetry axes for articulated and highly distorted shapes.

4.1 Introduction

The idea of self-similarity of contours can be derived from the notion of good-continuation formulated by Gestalt Psychologists (Wertheimer [271]). Good continuation has usually been interpreted as representing smoothness of a contour, measured by a total curvature or curvature variation (e.g., Shaashua & Ullman [229]; Pizlo et al., [211].) However, examination of examples provided by Wertheimer [271] shows that good continuation was supposed to measure something more general than just curvature. Specifically, good continuation seemed to refer to symmetries of a contour: mirror, translational and rotational symmetry. In all these kinds of symmetry, one part of the contour is identical to another, except for translation, rotation or reflection transformations. For example, a sine wave has translational symmetry, whereas a square has mirror and rotational symmetries. But contours in a shape are almost never perfectly symmetric. Therefore, in order to find symmetry in a shape, instead of verifying identity of parts of contours, one should evaluate their *similarity*. It follows that a contour can be called approximately symmetric when it is self-similar. In particular, the conventional examples of good continuation, such as a straight line and a circle, are self similar. In contrast to the methods used for mathematically symmetric shapes, we introduced a more general concept, which is an axis of self similarity. It also applies to non-rigid shapes, shapes with skewed symmetry, and performs well in the presence of contour noise.

It has been shown in Latecki & Lakamper [169] that a small set of critical points is sufficient to describe any planar shape in accord with human visual perception. Hence we measure self-similarity using a set of critical points on the contour. We employ the Discrete Curve Evolution method (DCE) ([169]) to compute the sets of critical points. Since the sets of critical points are very small (usually not exceeding 20 points) and since they capture all the information required for computing similarities, we are able to maximize self similarity by exhaustive search among axes of similarity. There have been many studies on shape similarity between objects based on their contours. In particular, Belongie, et al. [18] introduced shape context where shapes are represented using distance and angular histograms of the contour points and histogram-distance measures are used to obtain similarity between the shapes. They are also able to obtain correspondences between point sets on the two contours. Although they can obtain good results, they fail in the presence of articulations (deformations which are perception invariant), since they use Euclidean distance between points for distance histograms. In order to address this problem, Ling and Jacobs [176] replaced the Euclidean distance with geodesic distance, which is called inner distance and very related to skeleton matching [14]. The results of inner distance are, as expected, much better than shape context for articulated shapes. However, since they use tangent angles, the inner-distance is not stable in the presence of significant contour noise. Similar to [176], several approaches have been developed to address the problem of non-rigid shape matching and finding intrinsic symmetries, such as [34, 35, 36].

We introduce a shape representation and a shape similarity measure inspired by [176] and use it to compute main self-similarity axis. Since tangent angles are sensitive to contour noise, we use the histograms of only the geodesic distances between points. This has suitable properties for identifying self-similarities. The related work is briefly introduced in section 4.2. Section 4.3 describes the way we simplify the contour using critical points. The details of the self similarity maximization using geodesic distances are described in Sections 4.4 and 4.5. Finally, in Sections 4.6 experimental results are presented.

4.2 Related Work

The idea of using shape similarity to measure approximate symmetry is appealing, and has been proposed in the literature. Zabrodsky et al. [282], used contour shape similarity to detect symmetry. They explicitly apply each symmetry transformation to a given contour, for example 90° rotation, and then check whether the transformed contour and the original contour are similar. In the proposed approach, we do not explicitly compute any symmetry transformations but instead maximize self-similarity directly by subdividing the contour cleverly. This allows us to detect generalized symmetry axes as shown in Fig. 4.5, which are not detectable when applying mathematical symmetry transformations, e.g., rotations or mirror symmetry, to the input shapes. Geiger, et al. [102] also use contour similarity to find symmetry: They start with a 2D shape, its boundary contour, and two different parameterizations for the contour (one parametrization is oriented counterclockwise and the other clockwise). To measure its self-similarity, the two parameterizations are matched to derive the best set of one-to-one point-to-point correspondences along the contour. They use skeleton graphs to guide contour similarity computation, but the symmetry computation is still based on parameterized contour similarity. Many 3D objects in real world, man-made and natural, are bilaterally symmetric. Clearly, the symmetry is not perfect in the mathematical sense, but one could easily detect it if provided with good approximations of their surfaces [35, 215]. Our goal is to recover symmetry from 2D shapes, but when 3D shapes projected onto a 2D space, the bilateral symmetry is generally distorted. Consider approximately planar surfaces of bilaterally symmetric 3D object that are themselves bilaterally symmetric. For such surfaces, assuming orthographic projection, viewed as an approximation to perspective projection, the shape contour exhibits skewed symmetry. The symmetry axis of the distorted shapes is the projection of the symmetry axis of the bilaterally symmetric planar surface of a 3D object. The symmetry axis and the skew angle make up two parameters that completely determine a skewed symmetry of a given planar object. The skew angle is the angle between the symmetry axis and parallel lines of symmetry that join corresponding symmetric points of the contour. Kanade [140] showed that the parameters of skewed symmetry constrain the possible orientations of the surface of the 3D object. This fact inspired a large amount of work on detecting skewed symmetry, e.g., Friedberg [99], Ponce [213]. However, these methods do not yield reliable results since they try to directly recover the parameters of skewed symmetry. In contrast, since our approach maximizes self-similarity we detect skewed symmetries without the need to recover the parameters. Also our self-similarity computation is far more general in that we can find partitions such that object parts have similar shapes. For examples see Fig. 4.5.

4.3 Extracting Critical Points using DCE

The Discrete Curve Evolution (DCE) method was introduced in [169]. Contours of objects (shapes) in digital images are distorted by digitization noise and segmentation errors. Hence it is desirable to eliminate the distortions while at the same time preserve the perceptual appearances sufficient for object recognition. DCE accomplishes this goal by treating the contour as a polygon and recursively removing least relevant vertices according to a well-motivated measure. This process is illustrated in Fig. 4.1, where the red lines illustrate the simplified polygons. In each evolutional step, a pair of consecutive line segments S_1, S_2 is replaced by a single line segment joining the endpoints of $S_1 \cup S_2$. The key property of this evolution is the order of the substitution. The

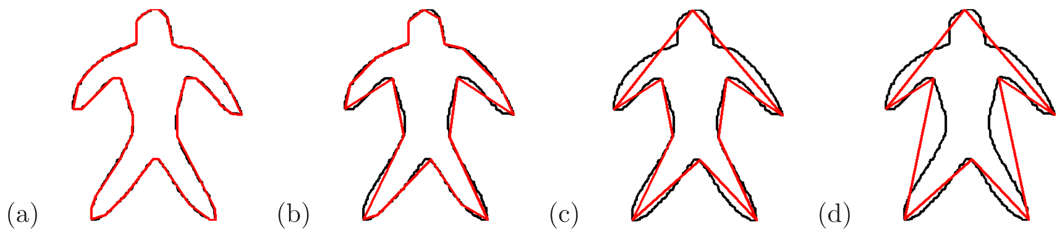
(a) (b) (c) (d)


Figure 4.1: The evolution of the Discrete Curve Evolution. The red lines are simplified polygons and the blue lines are the original contours. From (a) to (d), the polygons become more and more simple.

substitution is achieved according to relevance measure K given by:

$$K(S_1, S_2) = \frac{\beta(S_1, S_2)l(S_1)l(S_2)}{l(S_1) + l(S_2)} \tag{4.1}$$

where line segments S_1, S_2 are the sides of the polygon incident at a vertex $v, \beta(S_1, S_2)$ is the turn angle at the common vertex of segments S_1, S_2, and l is the length function normalized with respect to the total length of a polygonal curve C. The main property of this relevance measure is explained in [169]. The higher the value of $K(S_1, S_2)$, the larger is the contribution of the arc $S_1 \cup S_2$ to the shape. Therefore, the process eliminates the less important points while keeping the important points. In our approach, we obtain contour divisions using the set of critical points given by the vertices of the DCE simplified polygon.

4.4 Maximizing Self-Similarity

The intuition is that we divide a given shape into two parts, and compute their dissimilarity value. The parts that minimize the dissimilarity value, i.e., maximize self-similarity, are used to define a main similarity axis, which for many shapes corresponds to the main axis of symmetry. Clearly, for simple geometric objects like a ball, the main axis of symmetry (as well as the main similarity axis) is not defined, since it is not unique. However, for most shapes it is unique, and the proposed approach is able to determine the main axis of symmetry. Also some objects exhibit more than one axis of symmetry (e.g., the letter x). The proposed approach can be easily extended to handle such cases. For simplicity of presentation we describe the computation of a unique, main similarity axis.

The shape is divided into pairs of sub-parts using the critical points obtained by Discrete Curve Evolution. For a given shape, let $V = (v_1, \ldots, v_M)$ be the critical points. We assume a clockwise direction of contour traversal. Any two indices $i, j \in \{1, \ldots, M\}$ such that $i + 1 < j(mod\ M)$ and $j + 1 < i(mod\ M)$ define four divisions into two possibly overlapping subsequences of V with the cyclic order *mod* M:

$$1.\ (v_i,\dots,v_j),(v_{j+1},\dots,v_{i-1});\quad 2.\ (v_i,\dots,v_j),(v_j,v_{j+1},\dots,v_{i-1},v_i)$$
$$3.\ (v_i,\dots,v_j),(v_j,v_{j+1},\dots,v_{i-1});\quad 4.\ (v_i,\dots,v_j),(v_{j+1},\dots,v_{i-1},v_i)$$

A pictorial illustration of the four divisions is shown in Fig. 4.2.

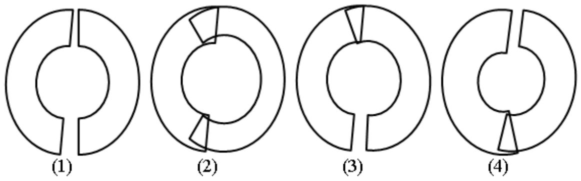


Figure 4.2: Pictorial illustration of the four divisions: (1) No common endpoint. (2) Two common endpoints. (3) and (4) One common endpoint.

Any of the four sequence divisions determines a shape division into two (possibly overlapping) contour segments relative to the two endpoints v_i, v_j. (Two ends of the contour segments can overlap only in one critical point.) For example, let us consider the convex critical endpoint pair 1, 3 in Fig. 4.3(a). It determines the following sequence divisions, and their corresponding sub-contours:

$$(1)\ (1,2,3),\ (4,5);\quad (2)\ (1,2,3),\ (3,4,5,1);\quad (3)\ (1,2,3),\ (3,4,5);\quad (4)\ (1,2,3),\ (4,5,1)$$

Two simplified sub-contours which are induced by sequence division (4) are shown in Fig. 4.3(b).

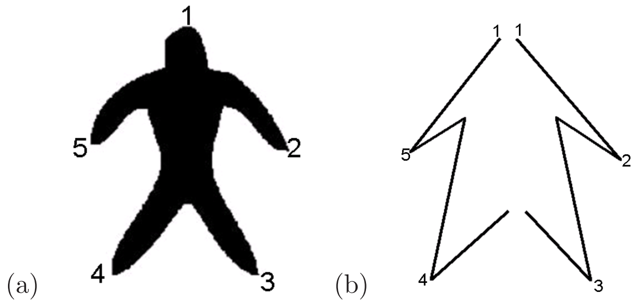


Figure 4.3: (a) The contour endpoints. (b) The sub-contour division induced by endpoint sequence division (1,2,3), (4,5,1).

For a given sequence division $(v_i,\dots,v_j)$, $(v_{j+1},\dots,v_{i-1})$, (we consider division pair (1) first), we define an induced division of the contour G into two possibly overlapping sub-contours S and T. The sub-contour S is defined as the union of critical points $(v_i,\dots,v_j)$. Similarly, we define the sub-contour T with respect to $(v_{j+1},\dots,v_{i-1})$. Observe that the corresponding sequences are exactly the vertices of S and T. While the order of endpoints in S is determined by the first sequence order, the order of endpoints in T is determined by the second sequence in reversed order. This means that the critical points $(v_i, \dots, v_j)$ of S are traversed clockwise while the critical points $(v_{j+1}, \dots, v_{i-1})$ of T are traversed counterclockwise. The same definition of the pair of sub-contours S and T applies to division pairs (2), (3), and (4).

Given a pair of induced sub-contours S, T, we compute their dissimilarity $c(S, T)$, where c is defined in Section 4.5, below. By enumerating all valid pairs of critical points indices $i, j \in \{1, \dots, M\}$ and the four possible sub-contours for each index pair, we find a pair of sub-contours S, T with the minimal dissimilarity $c(S, T)$. The obtained sub-contour division then defines the main self similarity axis. For symmetric and nearly symmetric shapes, the main self similarity axis corresponds to the main axis of symmetry.

Observe that the proposed approach is guaranteed to find the global minimum of self similarity of the contour of every planar shape, since we maximize the self similarity in the discrete domain determined by the index pairs. Also the number of dissimilarity comparisons needed is $O(M^2)$. Usually M is a very small number: even for complex shapes it is typically not larger than 20.

Since we do not know the optimal number of DCE vertices to best represent a given shape, we generate a range of 16 possible representation levels from coarse (5 vertices) to fine (20 vertices). At each representation level we find the self-similarity axis as explained above. The final symmetry axis is the best self-similarity axis chosen from across the 16 levels.

4.5 Self-Similarity Measure

Each division constructed in the previous section produces a pair of sub-segments of a given contour. In this section we explain the similarity measure used to compare two sub-segments of the contour. Our similarity measure is motivated by inner distance introduced in [176]. The advantage of geodesic distance is that it is insensitive to the articulation of parts, which is very important for shape similarity. In Fig. 4.4, the hand on the right is distorted, but the distance of the shortest path between the two marked points is still similar to that of the undistorted hand on the left.

Compared to geodesic distance, the Euclidean distance does not have this property. The reason for this is the Euclidean distance does not consider whether the line segment crosses shape boundaries. Based on the above discussion, even though the shape itself is distorted, the inner-distance descriptor is able to represent the shape correctly. Therefore, the proposed method uses inner-distance instead of Euclidean distance. However, since the proposed approach is for obtaining symmetric division within the same shape, we adapt it so as to be more suitable for measuring *self-similarity* of shapes. For each sub-part we compute a vector of geodesic distances between the contour points in that part. The order is determined using critical points that actually induce divisions. The reasons for discarding the angular information are (1) it is sensitive to contour noise and (2) it is not so discriminative for symmetry based on self-similarity. Observe that we use vectors

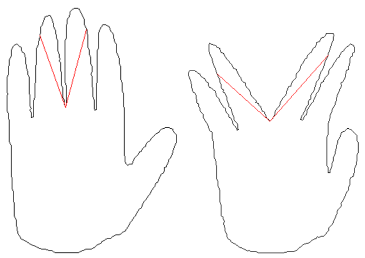


Figure 4.4: The red lines denote shortest paths within the shape boundary that connect pairs of contour points.

of geodesic distances instead of distance histograms. Instead of computing the vectors for each division separately, we first compute the distances and store them. We keep the order of points in two directions (clockwise and counter clockwise) when we calculate the geodesic distance from the current point to all other points. Therefore, for a shape which has N sample points $\{x_1, x_2, \ldots, x_{N-1}, x_N\}$ and $M << N$ critical points $\{v_1, v_2, \ldots, v_{M-1}, v_M\}$ chosen from the N sample points using DCE, there will be two distance matrices, for clock-wise and counter clockwise directions. For each critical point v_i, the clockwise distance vector to all N sample points is given by:

$$D(v_i) =< d(v_i, x_k), d(v_i, x_{k+1}), \ldots, d(v_i, x_{k+N-2}), d(v_i, x_{k+N-1}) > \tag{4.2}$$

where, x_k is the closest sample point to v_i and $k + m = (k + m \mod N)$ and $d(v_i, x_k)$ is the value of geodesic distance between the points v_i and x_k.

The counter-clockwise distance vector of v_i is given by flipping the sequences in eq. (4.2):

$$D'(v_i) =< d(v_i, x_{k+N-1}), d(v_i, x_{k+N-2}), \ldots, d(v_i, x_{k+1}), d(v_i, x_k) > \tag{4.3}$$

The proposed method has three main steps for the shape similarity. First for two vertices v_i and v_j, which are from different sets, the distance between them is calculated by

$$c(v_i, v_j) = \|D(v_i) - D'(v_j)\| \tag{4.4}$$

Then for two different sub-contours S and T (Section 4.4), with critical points $[v_i]_{i=1}^{M_1}$ in clockwise direction and $[v_j]_{j=1}^{M_2}$ in counter clockwise direction, we compute all distances between the points by formula (4.4) and obtain a distance matrix:

$$c(S,T) = \begin{bmatrix} c(v_1, v'_1) & c(v_1, v'_2) & \cdots & c(v_1, v'_{M_1}) \\ c(v_2, v'_1) & c(v_2, v'_2) & \cdots & c(v_2, v'_{M_1}) \\ \vdots & \vdots & \ddots & \vdots \\ c(v_{M_1}, v'_1) & c(v_{M_1}, v'_2) & \cdots & c(v_{M_1}, v'_{M_1}) \end{bmatrix}$$

Finally we compute the distance $c(S,T)$ between sub-contours S and T as the shortest path in matrix $C(S,T)$ from $c(v_1, v'_1)$ to $c(v_{M_1}, v'_{M_1})$ using dynamic programming [52]. It is equivalent to computing the dissimilarity value $c(S, T)$ by the optimal matching of the critical points from S to T with dynamic time warping.

4.6 Experimental Results

In this section we present the results of the described approach on a set of shapes from (Aslan and Tari [12]). In [12], the authors introduced a shape matching algorithm based on the skeleton of shape. Though the method can obtain excellent results for shape similarity, it seems hard to extend it for self-similarity. The dataset exhibits large shape variance due to distortion and articulation within each class, composed of four shapes. The results in Fig. 4.5 demonstrate that the proposed method is stable for articulated shapes. The red lines and blue lines represent two different shape symmetric sub-contours for each shape. Observe that the proposed method can obtain correct shape symmetric sub-contours for articulated shapes and it is also stable for obviously symmetric shapes as shown in Fig. 4.5. Moreover, the results in Fig. 4.6 show that even if the shape

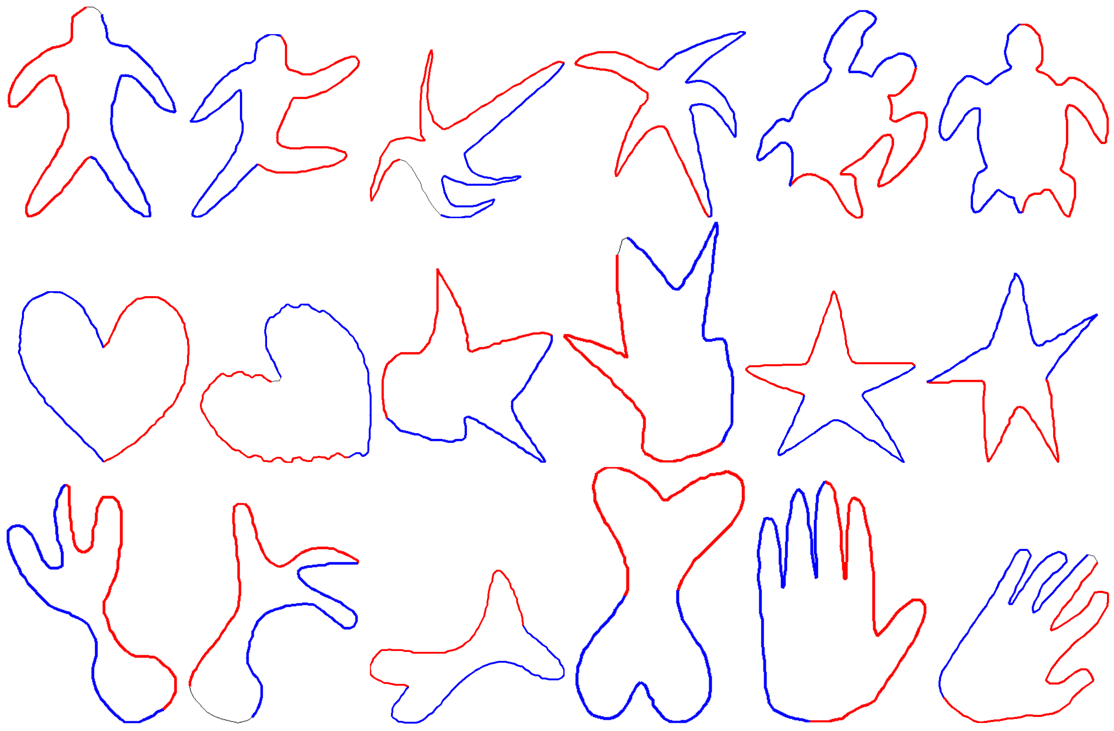


Figure 4.5: The symmetric results for articulated objects, The red lines and blue lines represent the symmetric sets. The gray lines are the lines which are skipped by the proposed method.

is not naturally symmetric, the proposed method can find a shape-symmetric division into two sub-contours which corresponds to a human's intuition. For example, the head of the horse is symmetric to tail and the two front legs are symmetric to the two rear legs. We demonstrated that the proposed

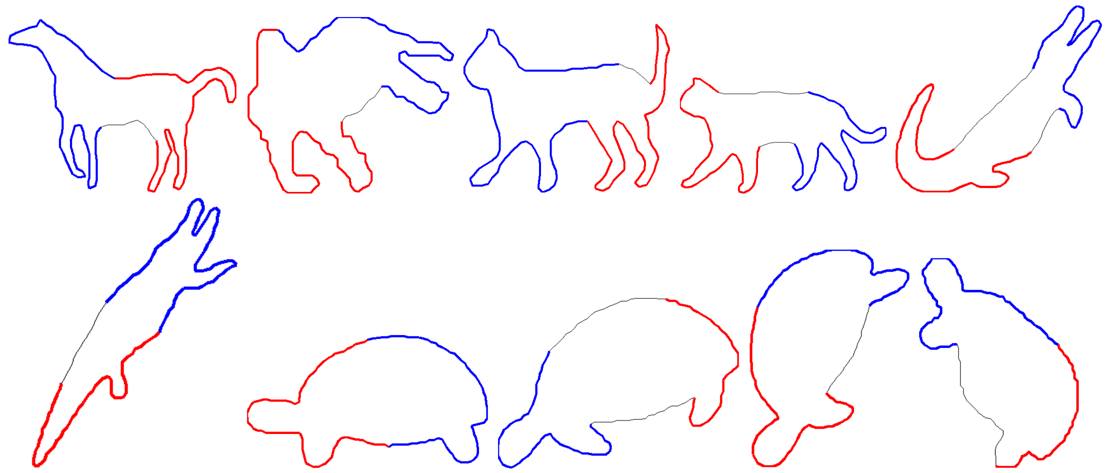


Figure 4.6: The symmetric results for objects which are not naturally symmetric. The red lines and blue lines represent the symmetric sets. The gray lines are the lines which are skipped by the proposed method.

method is robust for calculating the symmetric division of articulated and distorted shapes. Now we show that it is also useful for determining skewed symmetry, which is very important for detection of 3D mirror symmetry. As stated in Section 4.2, the symmetry axis of the skewed symmetry is the orthographic projection of the symmetry axis of the bilaterally symmetric planar surface of a 3D object. We illustrate in Fig. 4.7 that the proposed method can be used to recover the main axis of skewed symmetry. Certainly, if the shape is too skewed, the proposed method cannot find

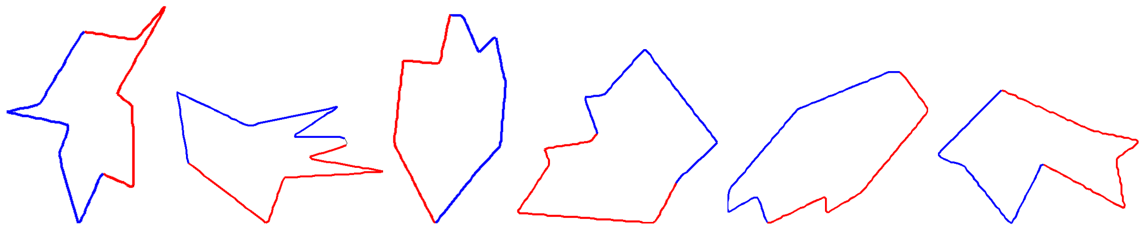


Figure 4.7: The symmetry results on skewed symmetric shapes. The red lines and blue lines represent the symmetric sets. The gray lines are the lines which are skipped by the proposed method.

the correct symmetric division. For example, the results in Fig. 4.8 show some incorrect divisions. However, examination of the shapes in Fig. 4.8, suggests that even humans may have problems identifying the axis of skewed symmetry.

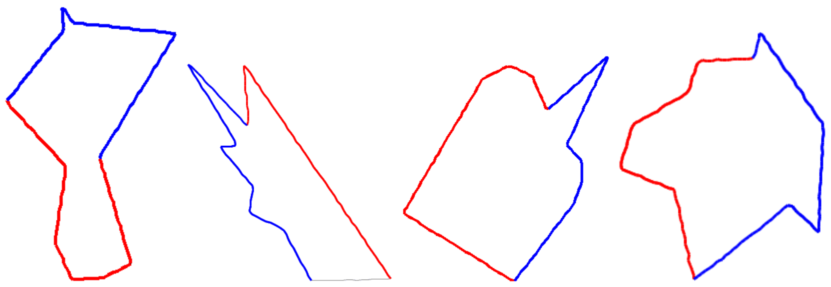


Figure 4.8: The wrong symmetry results in the presence of strongly skewed symmetric shapes. The red lines and blue lines represent the symmetric sets. The gray lines are the lines which are skipped by the proposed method.

4.7 Conclusions

We proposed a simple and novel method for computing symmetry of a shape that yields results in accordance with human intuition. We use vectors of geodesic distances to exploit the order of contour points for self-similarity. The experimental results show that the proposed method obtains correct shape symmetry division not only for articulated shapes, but also for skewed symmetric shapes, which is very important in determining the symmetry of 3D objects.

Acknowledgments

We thank Tadamasa Sawada for providing the dataset used for identifying the axis of skewed symmetry. This work was supported in part by the NSF Grants IIS-0534929, IIS-0812118, and by the DOE Grant DE-FG52-06NA27508.

CHAPTER 5

Grouping Object Boundaries using Hierarchical Models

In this chapter we present a novel framework for contour based object detection from cluttered environments. Given a contour model for a class of object, it is first decomposed into fragments hierarchically. Then, we group these fragments into *part-bundles*, where a part-bundle contains overlapping fragments. Using this model, we develop an efficient voting method using local shape similarity that generates high quality candidate part-configurations. To find optimal configuration, we use global shape similarity for false positive pruning. Furthermore, we show that appearance information can be used for improving detection for objects with rich texture patterns (e.g., giraffes). In the detection experiments on the ETHZ shape classes, our approach outperforms all previously reported methods.

5.1 Introduction

The key role of contours and their shapes in object extraction and recognition in images is well established in computer vision and in visual perception. Extracting edges in digital images is relatively well-understood and there are robust detectors like [188, 100]. However, it is often difficult to distinguish edge pixels corresponding to meaningful object contours. The main problem is that usually most edge pixels represent background and irrelevant texture, and only a small subset of edge pixels corresponds to object contours. Further, the edge pixels do not simply form occluding contours but broken contour fragments due to noise and occlusion. Grouping edge pixels into contours using various saliency measures and cues has been studied for long and is still an active research field [285, 253, 244, 124, 242].

Once contours are identified they can be further grouped into objects by performing shape matching with model contours. For example, Shotton et.al. [230] and Opelt et.al. [207] use Chamfer

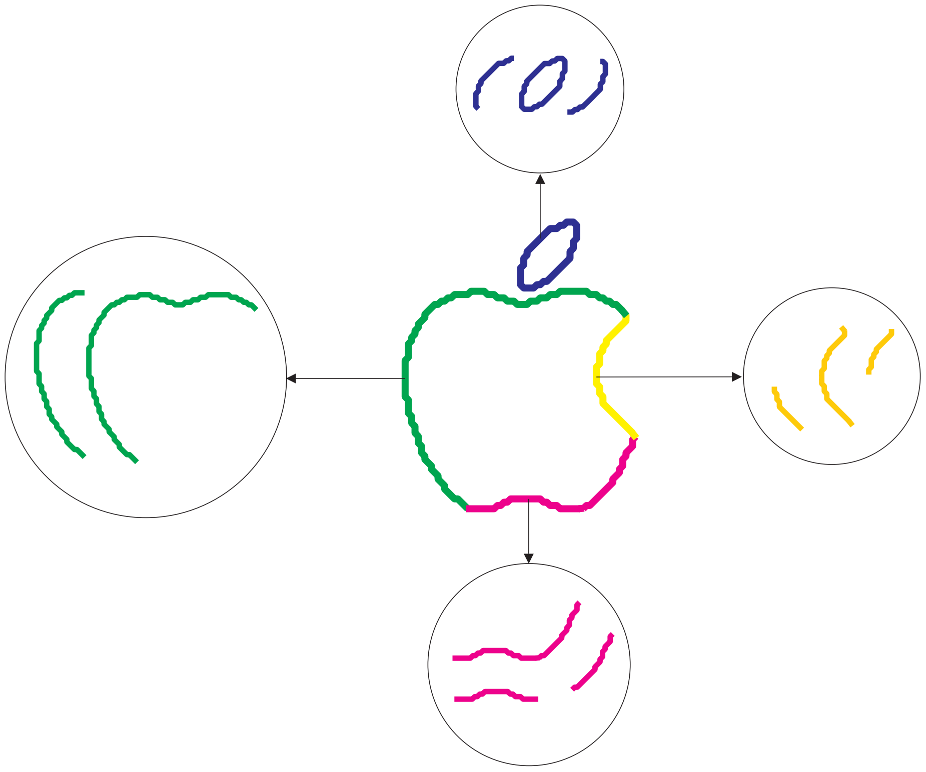


Figure 5.1: The contour model of the apple and the corresponding part bundles. The contour is shown in the center. All the fragments are decomposed to four part-bundles. Different contour parts and the corresponding bundles are shown in different colors.

distance [30] to match fragments of contours learnt from training images to edge images. McNeil and Vijayakumar [190] represent parts learnt from semi-supervised training as point-sets and establishes probabilistic point correspondences for the points in edge images. Ferrari et.al. [86] use a network of nearly straight contour fragments and sliding window search. Thayananthan et.al. [247] modify shape context [18] to incorporate edge orientations and Viterbi optimization for better matching in clutter. Flezenswalb and Schwartz [84] presents a shape-tree based elastic matching among two shapes and extended it to match a model and cluttered image by identifying contour parts (smooth curves) using [83]. More recently, Zhu et.al. [286] formulate the shape matching of contours (identified using [285]) in clutter as a set-set matching problem. They present an approximate solution to the hard combinatorial problem by using a voting scheme ([266, 174]) and a relaxed context selection scheme using linear programming. They use shape context [18] as shape descriptor. Aside from the recent work mentioned above, there are many early studies that use geometric constraints for model-based object and shape matching [114, 113, 115].

In this paper we focus on object detection by grouping the edge fragments in an image according to a contour model. Like many previous studies [266, 174, 230, 286, 87], we follow a two-phase framework: (1) Generating object hypotheses, and (2) picking the best one. We made contributions to both phases. For the first phase, we propose a novel *part-bundle* model that allows us to use local shape similarity of contour parts to efficiently generate accurate hypotheses. Specifically, we decompose a given contour model into several part-bundles, where each part-bundle contains several overlapping contour fragments (see Fig. 5.1 for an example). The model naturally handles hierarchical relations between fragments, as well as the AND/OR relations for co-existing and competing fragment pairs. By matching the fragments in the part-bundles to the edge fragments using their shape we can generate concise and accurate hypotheses for part-configurations. The part-bundle model is closely related to the hierarchical Markov Random Filed model [171] and the AND/OR graphs [121, 290, 49]. But our model avoids explicit graph-theoretic parsing while keeping all the necessary properties for inferring part-configurations.

In the second phase, we use global shape similarity with shape context [18] to compare candidate part-configurations to the whole model contour. Shape context is also used in [286] for similar purpose, but using only control points. In contrast, our shape context is more accurate since it is built on all contour points in both models and hypotheses. We can afford to do this due to the concise and accurate hypotheses generated in the first phase using local shape matching of edge fragments to the part-bundles. Another contribution we made for the second phase is to investigate how appearance helps for the task. Using the bag-of-features [284, 272, 260, 109] model, we show that the detection performance can be significantly improved for objects with rich texture patterns. The illustration of our framework is shown in Fig. 5.2.

The rest of the paper is organized as follows: In §5.2 we describe the first phase of hypotheses generation. §5.3 describes how we select the best hypothesis in the second phase. In §5.4 we present experimental evaluation of our method, followed by conclusion and discussions in §5.5.

5.2 Hypotheses Generation using Part-Bundles

The first stage of our object detection focuses on generating good candidate part-configurations of the edge fragments in the image, i.e., hypothesis generation. Given an input image I and its edge map, we first apply edge linking algorithms like [149, 285] to get the set of edge fragments, denoted as $E = \{e_i\}_{i=1}^{n}$. A sample set of edge fragments can be seen in Fig. 5.2(a). Let $\mathcal{B}$ denote our model (details of $\mathcal{B}$ are delayed to the following subsections), then our task is to find a candidate subset of E such that the global shape composed by it matches to that of the $\mathcal{B}$.

5.2.1 Part-Bundles Model for Shape Decomposition

To model an object class, we first get its contour representation by manually tracing an sample image as in [88, 286]. Then, we decompose it into to contour fragments using discrete curve

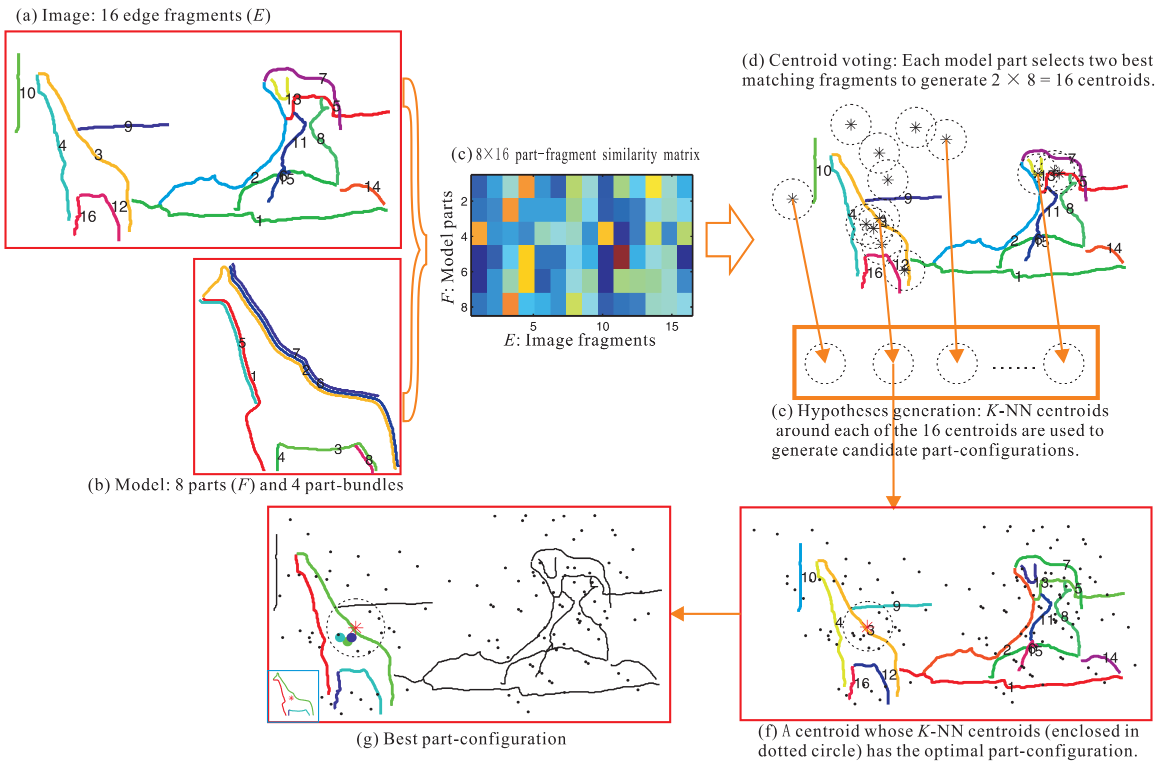

(a) Image: 16 edge fragments (E)
(b) Model: 8 parts (F) and 4 part-bundles
(c) 8×16 part-fragment similarity matrix
F: Model parts
E: Image fragments
(d) Centroid voting: Each model part selects two best matching fragments to generate 2 × 8 = 16 centroids.
(e) Hypotheses generation: K-NN centroids around each of the 16 centroids are used to generate candidate part-configurations.
(f) A centroid whose K-NN centroids (enclosed in dotted circle) has the optimal part-configuration.
(g) Best part-configuration

Figure 5.2: Illustration of our framework: (a) The edge fragments are generated using edge-linking software [149]. (b) Contour based model is decomposed into part-bundles (see §5.2.1). (c) & (d) Using part-fragment similarity we perform efficient centroid voting (see §5.2.2). (e)-(f) The part-configurations generated using K-NN centroids around each centroid are evaluated using shape and appearance information and the best part-configuration is picked (see §5.2.2 and §5.3).

evolution (DCE) [169]. Let $F = \{f_i\}_{i=1}^m$ denote the set of such fragments. Matching F and E is hard combinatorial *set-set* matching problem. [286] recently presented a context selection scheme in which they define control points on the fragments and then perform a *one-one* matching between the control points. They solve a relaxed version instead of the discrete version with linear programming.

In contrast we propose a *part-bundle* model to decompose F into a small group of bundles. A part-bundle B is a subset of F, such that at most one fragment in B is allowed to have correspondence to an edge fragment in E. In other words, all fragments in a bundle B compete with each other during shape-to-image matching. With this bundle decomposition, we now can write our contour model as

$$\mathcal{B} = \{B_k\}_{k=1}^{m'} \ ,$$

where m' is the number of part-bundles in model $\mathcal{B}$, which is usually very small. For bundles we have the following constraints

$$B_k \subset F \ , \qquad k = 1, ..., m'$$
$$B_i \bigcap B_j = \emptyset \ , \quad 1 \leq i < j \leq m' \ .$$

An example bundle decomposition is shown in Fig. 5.1. It can be seen that a bundle can have fragments representing overlapping parts thus having redundancy and hierarchy. A cognitive motivation behind such decomposition scheme is that an object can be recognized even if some parts of it are missing, as can be observed in Fig. 5.3. There are several reasons why parts of objects can be missing in real images: missing edge information, occlusion, failures in contour grouping. Our experimental results presented in §5.4 show the superiority of our part-bundle scheme over the control point scheme.

Taking advantage of the exclusion property of part-bundles, we can model the mapping from $\mathcal{B}$ to E as a one-to-one mapping. Denote such a mapping as $\pi : \{1, ..., m'\} \rightarrow \{0_1, 0_2, \ldots, 0_{m'}, 1, ..., n\}$, such that $B_i \in \mathcal{B}$ is mapped to edge fragment $e_{\pi(i)} \in E$ or empty edge when $\pi(i) = 0_i$[1]. It can be seen that each such mapping can be considered a part-configuration obtained by the $\{e_{\pi(i)}\}_{i=1}^{m'}$ in the image and our task is to find an optimal one $\widehat{\pi}$. Searching among all configurations is very expensive since there are $\prod_{i=0}^{m'-1}(n-i) = O(n^{m'})$ possible configurations.

5.2.2 Efficient Centroid Voting by Part-Bundles

As mentioned in the previous section, our part-bundle generates a very concise model decomposition, with $m' \approx 4$ for most objects. Taking advantage of this fact, we design an efficient voting scheme for searching in the configuration space.

The basic idea is to use fragment correspondences between F and E to vote for the object centroid. Specifically, for an image fragment $e \in E$ and a model fragment $f \in F$, we first build the mapping between the sample points. Using sequence of tangent directions along e and f as their shape descriptors we perform sequence matching using Smith-Waterman [268]. Then, using the part-centroid relation from the model, we can project the expected centroid of the object in the image. For each $f \in F$, we rank all fragments in E according to their similarities to f as computed above. Then we choose the best two matches for centroid voting. Consequently, the whole voting procedure generates $2m$ centroid votes (see for e.g. Fig. 5.2(c) & (d)).

The correspondences in the optimal configuration $(\widehat{\pi}(1), \ldots, \widehat{\pi}(m))$ should consistently estimate the centroid of the object. Assuming there exists a centroid whose K-NN centroids correspond to the optimal fragment-bundle correspondences, we just need to examine part-configurations obtained from K-NN centroids around all $2m$ centroids (for e.g. see Fig. 5.2(e)). A group of K-NN centroids gives us $\left(\frac{K}{m'}\right)^{m'}$ part-configurations. So we have to examine a total of $2m\left(\frac{K}{m'}\right)^{m'}$ hypotheses or part-configurations. We denote the set of these hypotheses as $\mathcal{H}$. The number of configurations

[1] 0_i denotes a dummy fragment that B_i maps to for handling missing parts.

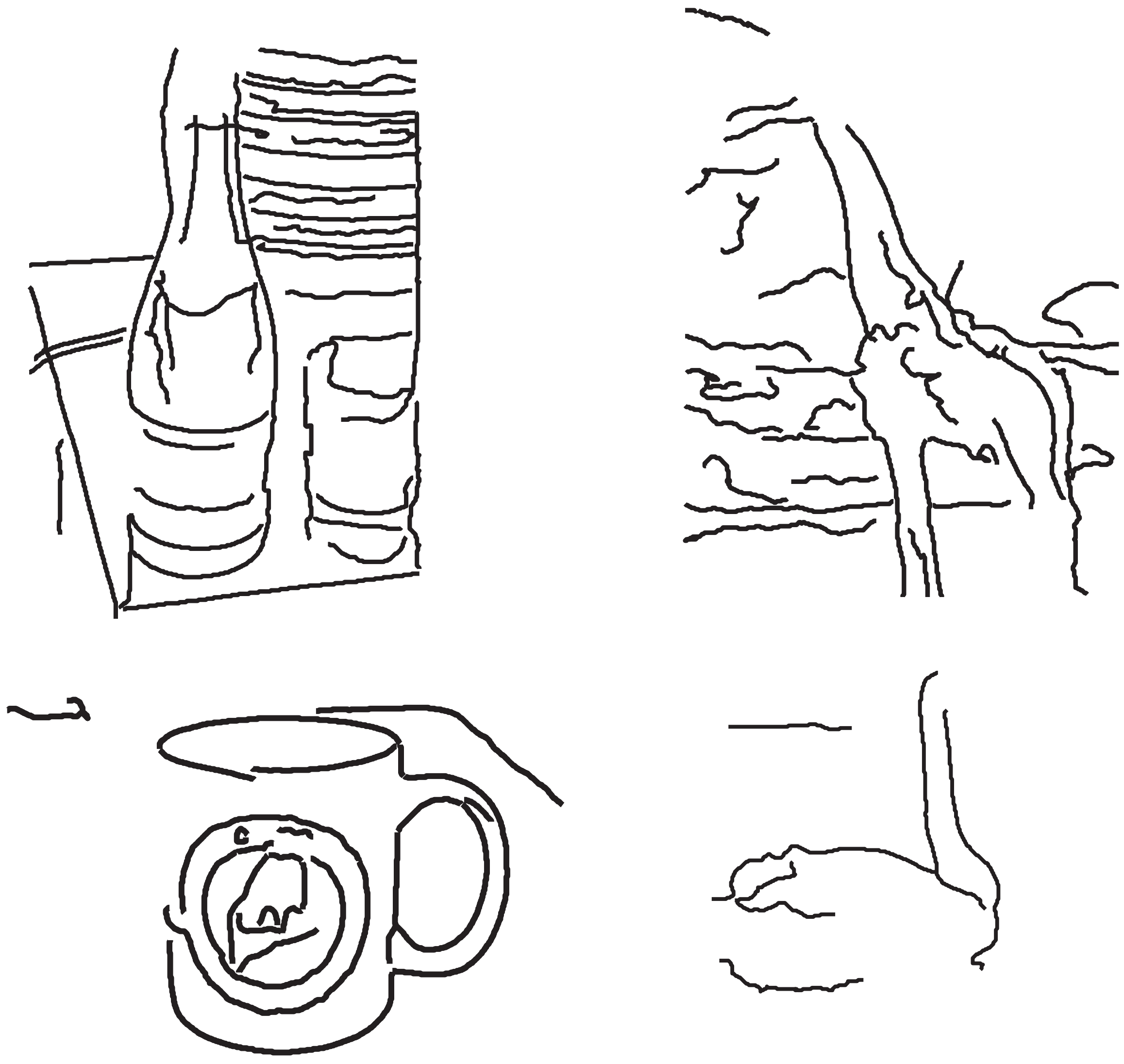


Figure 5.3: Parts of the objects are missed both due to missing edge information and due to broken edge links. Objects can still be recognized using shape even if some part-fragments are missing.

to be explored is exponential in the number of part-bundles but usually $m' \approx 4$ for most of the models. Also since $m << n$, $2m \left(\frac{K}{m'}\right)^{m'} << O(n^{m'})$ as long as $K << nm'$. We have empirically observed that $K = 20$ is a good selection.

Note that the proposed searching approach uses a brute force scheme, which provides more robust configuration estimation than traditional solutions that usually rely on mode seeking. Though more reliable, brute force solutions are usually forbidden due to their high computational complexity. The key factor that makes our solution affordable is the part-bundle model, which provides a very efficient model representation since m' is very small.

5.3 Hypotheses Evaluation

5.3.1 Using Local and Global Shape

Our part-configurations are groups of edge fragments and our models are based on contour parts. Hence it is natural to use shape constraints for evaluating the part-configurations. Instead of just using geometric relationship between contour parts, we use global shape similarity to judge the quality a candidate.

We perform a coarse to fine shape matching in evaluating the hypotheses. We first eliminate hypotheses $h_i \in \mathcal{H}$ whose cumulative part to fragment similarities measured using [268] is below a certain threshold. Then for the remaining hypotheses we use Shape Context [18] which is a well known global shape descriptor and has demonstrated excellent performance for shape analysis tasks. We use it to estimate the similarity between a model and a hypothesis. Specifically, let S be the shape (i.e. point sets) generated by the hypothesis h, and $S^{\mathcal{B}}$ be the model shape. The shape distance between them is defined by the their shape context distance ([18]) $d_{SC}(S, S^{\mathcal{B}})$. We then pick the best candidate according to d_{SC}.

5.3.2 Combining Shape and Appearance

In addition to shape, appearance information often provides distinctive cues for object detection and recognition, especially for objects with rich texture patterns. In this subsection, we describe our study along this line.

We use the popular bag-of-features [284, 260] framework for appearance analysis. First, local patch descriptors are generated from training images using SIFT [182], which are used to build a dictionary $\mathcal{D} = \{w_i\}_{i=1}^{M}$ using k-means. Second, all local features in a window are mapped to visual words in $\mathcal{D}$, and the window is then represented by its word frequency vector. Third, we collect all such word frequency to form a training set, and use support vector machine (SVM) [259] to learn a classifier.

For object classes with rich texture like giraffes we evaluate the hypotheses using appearance information in addition to shape. Again we first eliminate part-configurations whose cumulative part-fragment similarities is below a certain threshold. Then for each of the remaining hypotheses hypothesis h, we use the texture inside the smallest bounding box enclosing the fragments to get its visual word frequency representation. Then we measure its relevance to a class using the confidence of the learned SVM model, denoted as $d_{app}(h)$. Combining shape context and appearance similarity, we have the following weighted evaluation criterion:

$$d_{mix} = d_{SC} + \lambda \, d_{app} \tag{5.1}$$

where λ is the weight for appearance.

5.4 Experimental Results

We present results on the ETHZ shape classes ([88]). It has 5 different object categories with 255 images in total. All categories have significant intra-class variations, scale changes, and illumination changes. Moreover, many objects are surrounded by extensive background clutter and have interior contours.

5.4.1 Experiments using Shape

Similar to the experimental setup in [286], we use only the single hand-drawn models provided for each class. The part-bundles created from the contour models can be seen in Fig. 5.4.

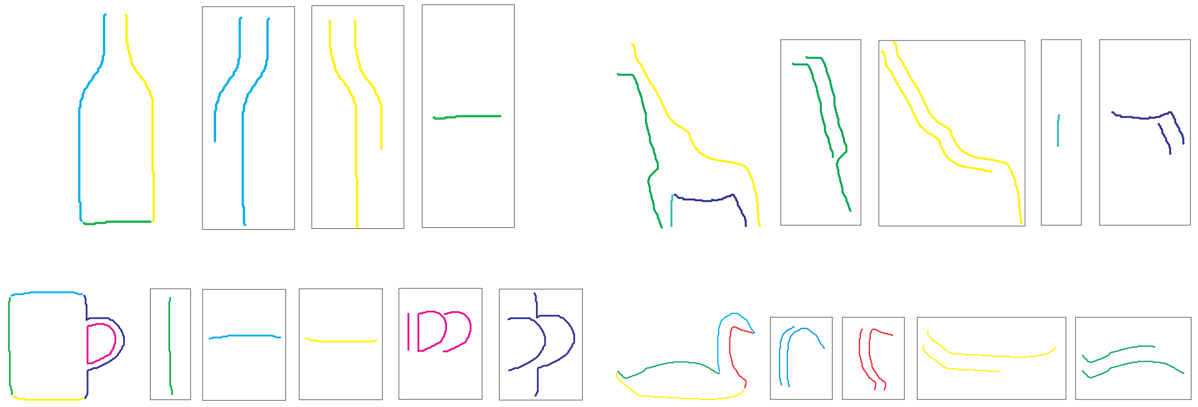


Figure 5.4: The part-bundles of the hand-drawn contour models of ETHZ shape classes. Different parts and the part-bundles are shown in different colors.

Precision vs. Recall (P/R) curves are used for quantitative evaluation. To compare with the results of [87] that are evaluated by detection rate (DR) vs. false positive per image (FPPI), we translate their results into P/R values as in [286]. As argued in [286], P/R is a more objective measure than DR/FPPI, because DR/FPPI depends on the ratio of the number of positive and negative test images and hence is biased. Fig. 5.5 shows P/R curves for the three methods: Contour Selection method of [286] in black, that of [87] in green, and our method in red. Our approach performs significantly better than the method in [87] on all five categories, and outperforms the Contour Selection method [286] on most categories.

We also present the precision to [286] and [87] at the same recall values in Table 5.1. The precision / recall pairs of [286] and [87] are quoted from [286]. Except for the class of Giraffes our precision is higher than that of [286].

On a PC with 2.3GHz Pentium (D) CPU and 3GB RAM, the average processing time of our method in MATLAB implementation per image is around 25 seconds.

Class (Recall%)	[87]	[286]	Our method
Applelogos (86.4%)	32.6%	49.3%	**62.5%**
Bottles (92.7%)	33.3%	65.4%	**66.0%**
Mugs (83.4%)	40.9%	25.7%	**42.0%**
Giraffes (70.3%)	43.9%	**69.3%**	63.5%
Swans (93.9%)	23.3%	31.3%	**59.0%**

Table 5.1: Precision / recall pairs of the method of [87] and the Contour Selection method of [286] are quoted from [286]. Our precision is higher than both the methods except for the class of Giraffes.

We use the same criterion for correct detection as in [87]: the detection is considered as a correct detection if the overlap between the detected bounding box and the ground truth bounding box is larger then 20%. Sample successful detections, failure cases and false-positives are shown in Fig. 5.6.

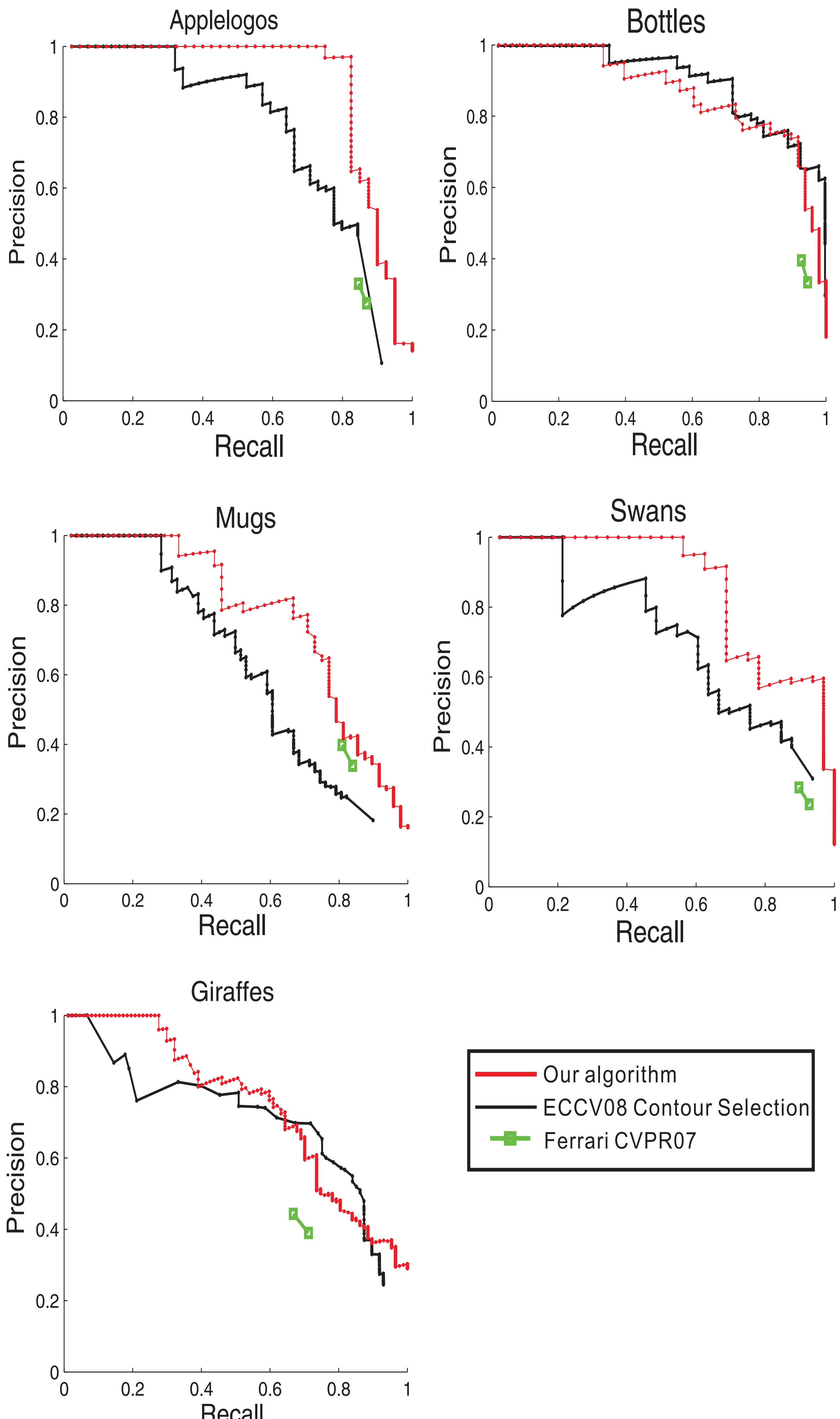
5.4.2 Experiments Combining Shape and Appearance

We now present results of using appearance information on the class of Giraffes which has rich texture pattern because of their skin. We pick the first 16 images from the class to obtain the code book $\mathcal{D}$ of 100 visual words. We only pick the sift features inside the ground truth bounding box. The word frequencies of the 16 images form the training set. Then, when detecting, each configuration finds its nearest neighbor according the distance d_{mix}, with $\lambda = 8$.

Fig. 5.7 shows the P/R curves using only shape and combining shape and appearance. Fig. 5.8 shows the improvement of using appearance information on two sample images. As can be seen using appearance information can significantly improve the detection performance.

5.5 Conclusion

In this chapter we studied the problem of contour-based object detection. A new contour model, part-bundles, is proposed that decomposes contour fragments into subsets. We show that this decomposition efficiently groups competing fragments and therefore enables brute force search hypothesis voting, which generates high quality and concise candidate detection sets. With these candidate sets, we use global shape similarity for false positive pruning and achieved excellent performance on ETHZ dataset in comparison with previous reported results. In addition, we show that appearance information which can be easily integrated into our approach can help achieving further improvement for objects with rich texture patterns.


Applelogos
Bottles
Mugs
Swans
Giraffes
Precision
Recall
0
0.2
0.4
0.6
0.8
1
Our algorithm
ECCV08 Contour Selection
Ferrari CVPR07

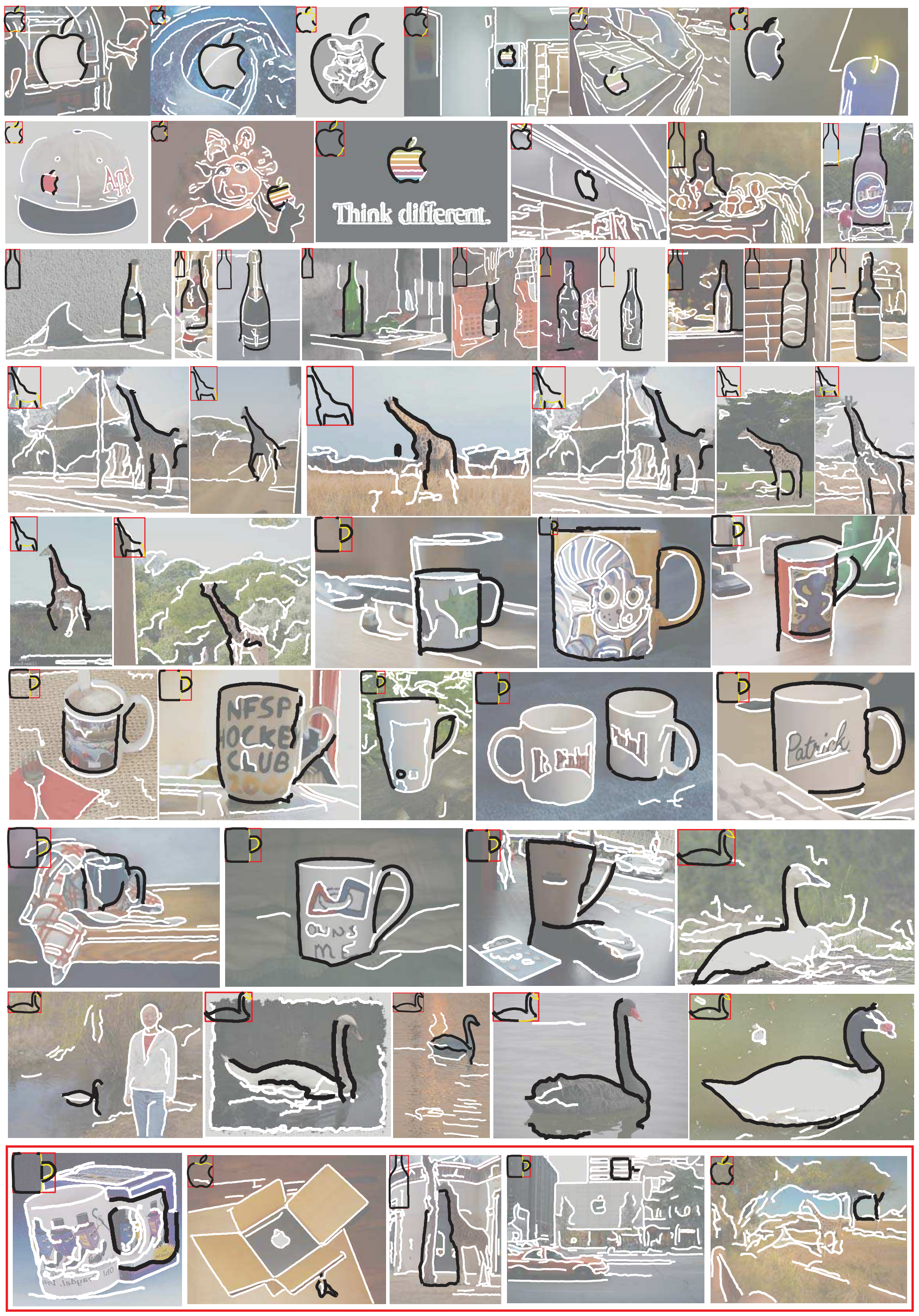


Figure 5.6: Sample detection results. The edge map is overlaid on the image in white. The detected fragments are shown in black. The corresponding model parts are shown on top-left corners. The bottom most row (enclosed in red) show some failure cases and false positives.

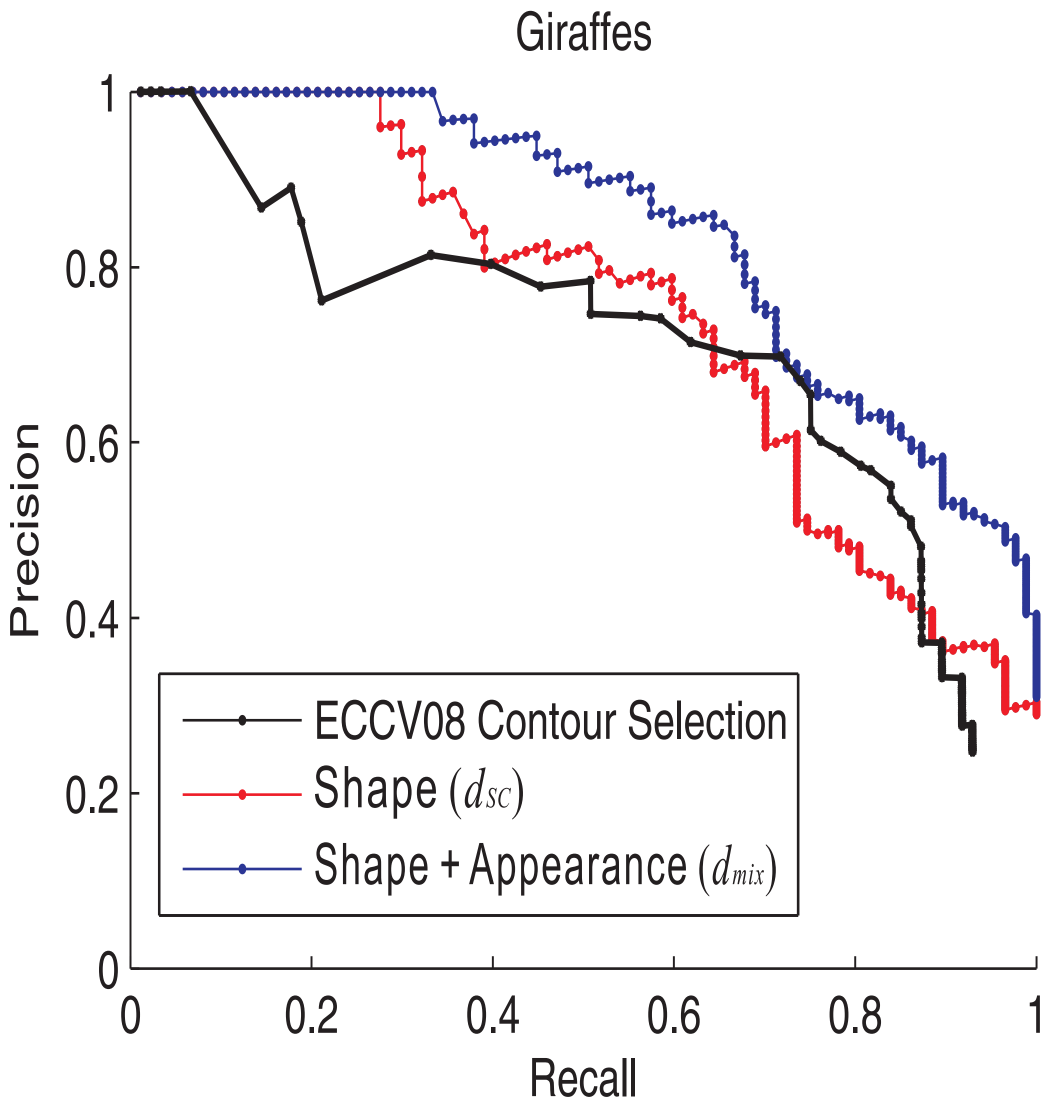


Figure 5.7: The Precision/Recall curves using shape based evaluation (d_{SC}) and combining shape and appearance (d_{mix}) on the class of Giraffes. We also show the results of [286] for comparison. The combination of shape and appearance significantly improves the results.

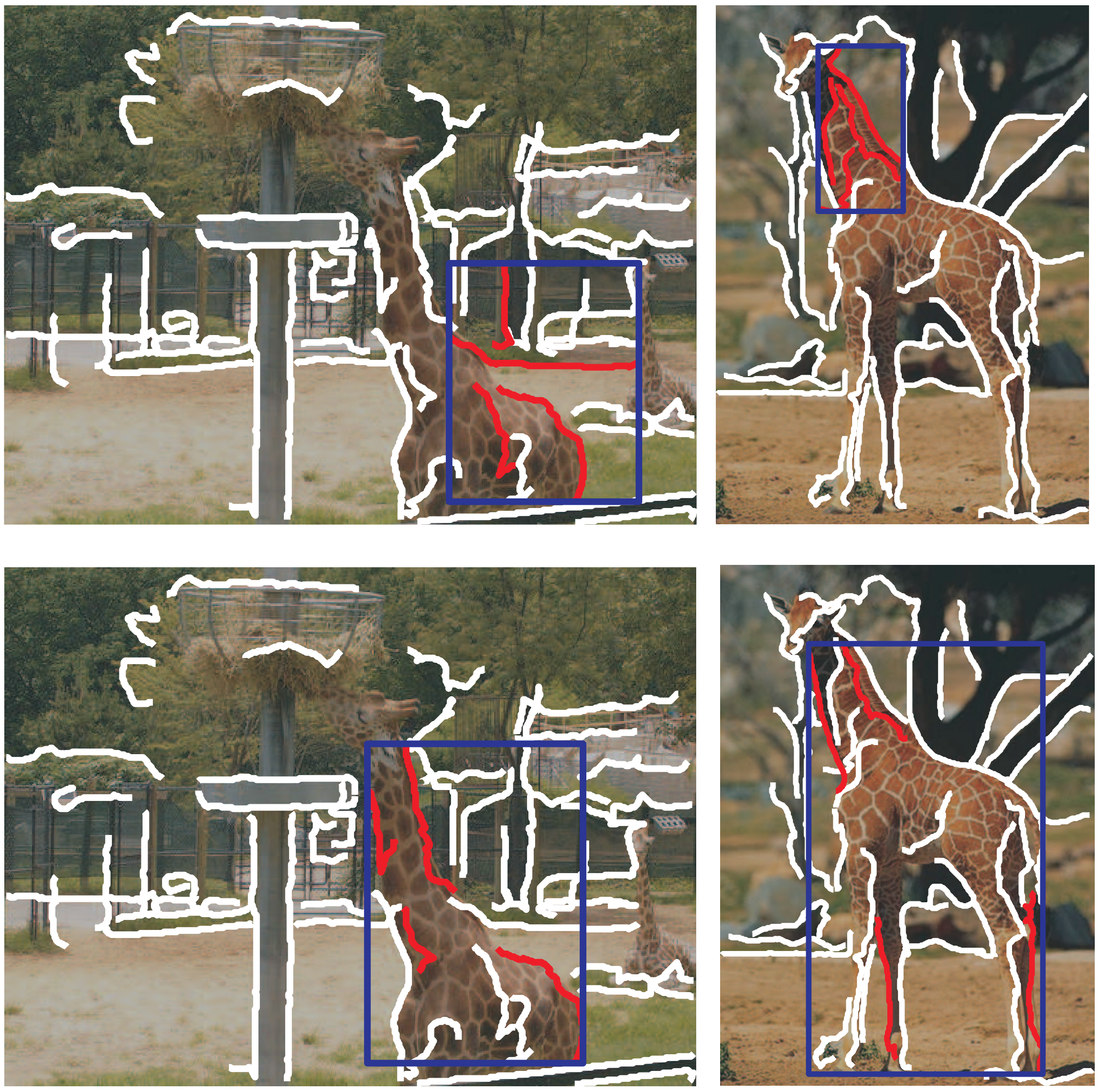


Figure 5.8: Two examples to show the improvement by combining appearance information. Top row: Results of using only shape. Bottom row: Results of combining appearance with shape.

Part II

Robot Mapping

CHAPTER 6

Shape based Estimate of Odometry for Mapping

In this paper we propose a robot mapping approach that is particularly suitable for rescue robots. Since rescue robots cannot rely on the odometry information, we use shape information extracted from range data of robot scans to find and align similar similar structures among different scans. We are able to build global maps of highly cluttered environments without any odometry information.

6.1 Introduction

The state of the art rescue robots are manually operated. A serious problem is that after certain movements the position of the robot may be lost and the task of the rescuer becomes very difficult. Therefore, a global map of the robot environment with a robot path marked can be very helpful to the rescuer. The overview knowledge in the form of a global map is particularly important to localize victims in catastrophe scenarios (e.g., in collapsed buildings) and to ensure that the whole target region has been searched [134]. Moreover, when a victim is found, the map gives a responder team a much better situation awareness than currently used sketches which are manually drawn. The process of creating a global map of the robot environment is called robot mapping.

Robot mapping in rescue scenarios is a very challenging task, because there are uneven surfaces often causing robots tilt. This implies, in particular, that the odometry information is very unreliable. Moreover,when mapping is done with a laser range scanner, the tilts can cause consecutive scans to be very different from each other because of possibly different structures in the environment getting scanned. The other challenge is that the rescue environments are usually highly cluttered.

The problems of self-localization, i.e. localizing the robot within its internal map, and robot mapping, i.e. constructing the internal map autonomously, are of high importance to the field of mobile robotics [249]. Coping with unknown or changing environments requires to carry out both tasks simultaneously, therefore this has been termed the SLAM problem: Simultaneous Localization and Mapping [59] and it has received considerable attention [59, 120, 249]. Successful stochastical approaches mostly based on particle filters have been developed that tackle representation and handling of uncertain data which is one key problem in SLAM [251]. As current stochastical models are powerful, even linking them to a simple geometric representation like reflection points measured by a range sensor already yields impressive results. Advances in stochastical means have improved the overall performance leaving the basic spatial representation untouched. As the internal geometric representation is a foundation for these sophisticated stochastical techniques, shortcomings on the level of geometric representation affect the overall performance.

The highly cluttered environment, possible robot tilt, and unreliable odometry information cause the existing SLAM approaches to fail, since they assume large overlaps between scans and also good estimates of robot poses. Also SLAM approaches fail in conditions where the environment changes rapidly like say in a crowded shopping mall or on campus due to occlusion.

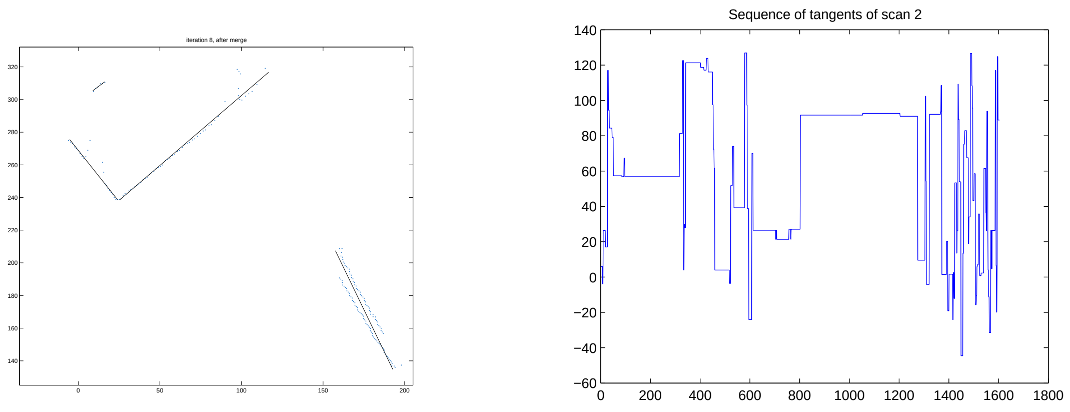
This paper presents an approach that makes it possible to successfully complete the task of robot mapping of highly cluttered environments without any odometry information. We use shape information extracted from range data of robot scans to find and align similar similar structures among different scans. Shape information abstracts from actual positions of scan points in local scan maps making it possible to identify similar structures without any assumption of large overlaps between consecutive scans. We only require that similar structures are present so that it is possible to identify common parts of the robot environment.

6.2 Matching structures using shape similarity

The scan data is a set of points obtained using range data. We abstract from the position of scan points by using the shape information obtained using *tangent directions* and *angular histograms*. We preserve the scan order and geometric properties of the scan and hence we can use the shape similarity to match similarities between scans.

6.2.1 Tangent Directions

The points on each scan are approximated by polylines using Expectation Maximization and perceptual grouping as discussed in [168]. For each point we find the closest polyline and its tangent direction is stored as the direction of the point. Since we maintain the order of scan points, we obtain a sequence of tangent directions for each scan. Approximation of point sets with polylines using EM is a robust technique, and hence the sequence of tangent directions represents robust shape information of a scan. Fig. 6.1 shows polyline approximation of points and the sequence of tangents obtained for a sample scan. Two points in different scans are considered similar if their associated tangent directions are within 10° shift.



(a) (b)

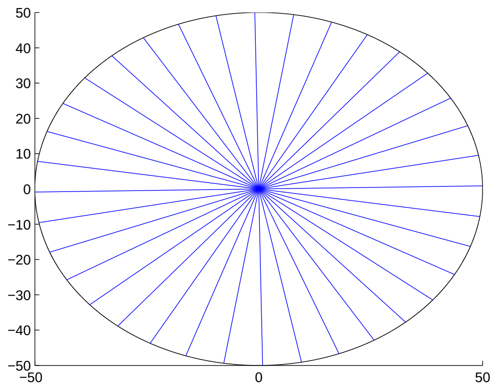
Figure 6.1: (a) shows a piecewise linear approximation of points in part of a sample scan obtained using EM. (b) A sequence of 1604 tangent directions.

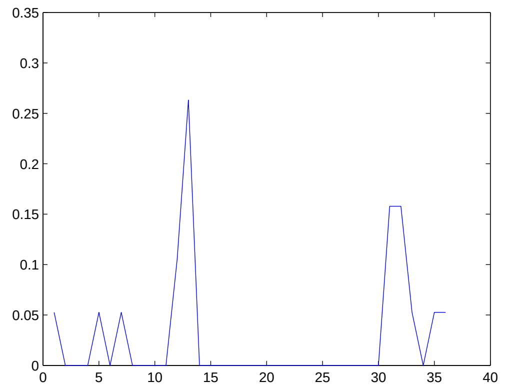
6.2.2 Angular Histograms

For each point in a scan we construct an angular histogram with 36 bins by counting the number of scan points for each bin within a radius of $50cm$. The circular frame and a sample angular histogram are shown in Fig. 6.2. This representation is similar to shape histograms used in [10] and to shape context used in [18]. But we use a different similarity measure. χ^2 statistic used in [18] is not applicable in our case, because in our application most bins are empty. We use the number of differing bins as the distance between two histograms. Two points in different scans are similar if they differ in at most 2 bins. We consider two bins b_1, b_2 as different if $|b_1 - b_2| > 0.25$ where b_i represents normalized value of the i^{th} bin.

6.2.3 Longest Common Subsequence

Once we obtain sequences of tangent directions and angular histograms between two scans, we use a well established algorithmic technique to obtain longest matching subsequences (LCS). LCS algorithm [201] considers two points to be similar based on similarity between both tangent directions and angular histograms as explained in Sections 6.2.1-6.2.2. This technique finds corresponding points between in two scans but there are two problems. One is in cases where there are no similar shape structures between two scans as illustrated in Fig. 6.3 and the other is false positives. The first problem can be solved by aligning the query scan to the partial global map composed of a few





(a) (b)

Figure 6.2: (a)Circular frame used for calculating angular histograms of points. (b)An example angular histogram.

recent scans instead of just the previous scan. To reduce computational demands we choose to align the query scan to a map composed of three previous scans. As shown in Fig. 6.4, this approach makes it possible to find similar structures.

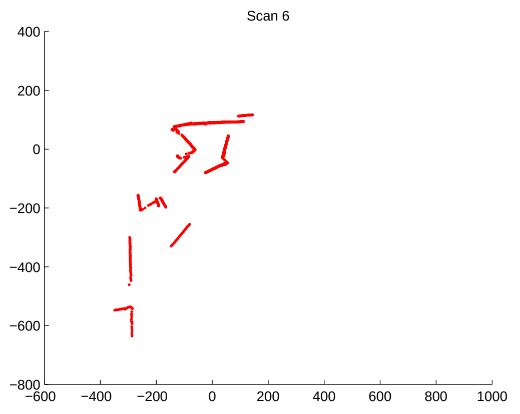
To understand the second problem we can view the matchings generated by LCS as the hypotheses of correspondences. We have to eliminate false hypotheses. The false positives are eliminated using two deterministic filters. These filters are explained in Section 6.2.4. To eliminate the problem of rotation of scans we perform the shape matching for several different rotations of the query scan. We chose the rotation that gives us the largest true overlap as the best rotation as defined in Section 6.3.

6.2.4 Eliminating false positive correspondences

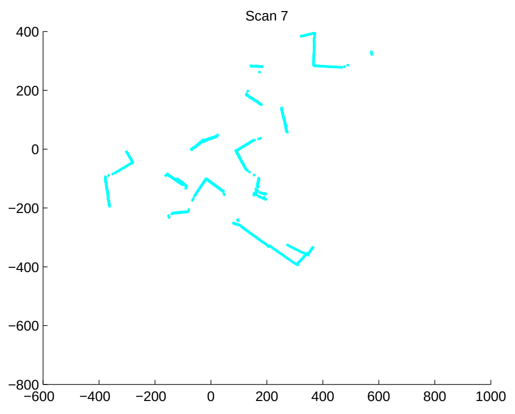
We eliminate false positives using two filters, namely colinear filter and segment filter.

- **Colinear filter:** The shape information of colinear structures is not very descriptive. This is why we eliminate colinear matches from the matches extracted by LCS. We use a very simple solution to the problem. If the direction of a correspondence vector is within $10°$ shift to the tangent direction of either of the corresponding points then the correspondence is considered colinear and removed.

- **Segment filter:** We first divide the LCS corresponding points into parts, called segments, and eliminate *bad* segments. Division of LCS into segments is based on simple jumps in the correspondences. If there is a jump of $30cm$ or more between one correspondence and the previous correspondence in either sequence then we create a new segment. For each segment



(a)



(b)

Figure 6.3: This figure shows two consecutive scans 6 and 7 in our test data set. Observe problem that there are no similar structures.

we compute a quality measure. Let the segment for which we compute the quality measure be called main segment. We first translate the query scan to the target scan based on the average translation vector calculated using the correspondences in the main segment. After putting the query scan onto the target scan we compute the average distance between the corresponding points in the main segment. Now we compute average distances for all other segments and eliminate those that are above twice the average distance for the main segment by $20cm$. Thus we eliminate only very bad segments. The quality measure of the main segment is the number of the left over corresponding points. We then pick the segment with highest quality measure which leaves us with LCS correspondences that are good after translation. The effect of eliminating false positive matches obtained by LCS can be seen in Fig. 6.5.

6.3 Robot Pose Estimation

We compute best translation and rotation to get good robot pose estimates. For each rotation we pick the best segment wise translation, score the rotation using *scan overlap measure* described below. Then we select the best rotation and the corresponding translation.

- **Segment wise translation** After we eliminate false positive correspondences we choose the best translation and rotation of query scan points onto model scan points. We use the segments computed in segment filtering. For each of the segments among the segments left after filtering we compute translation and put the query scan onto the target scan. We then compute the average distance between the correspondences in the segment used for translation. We

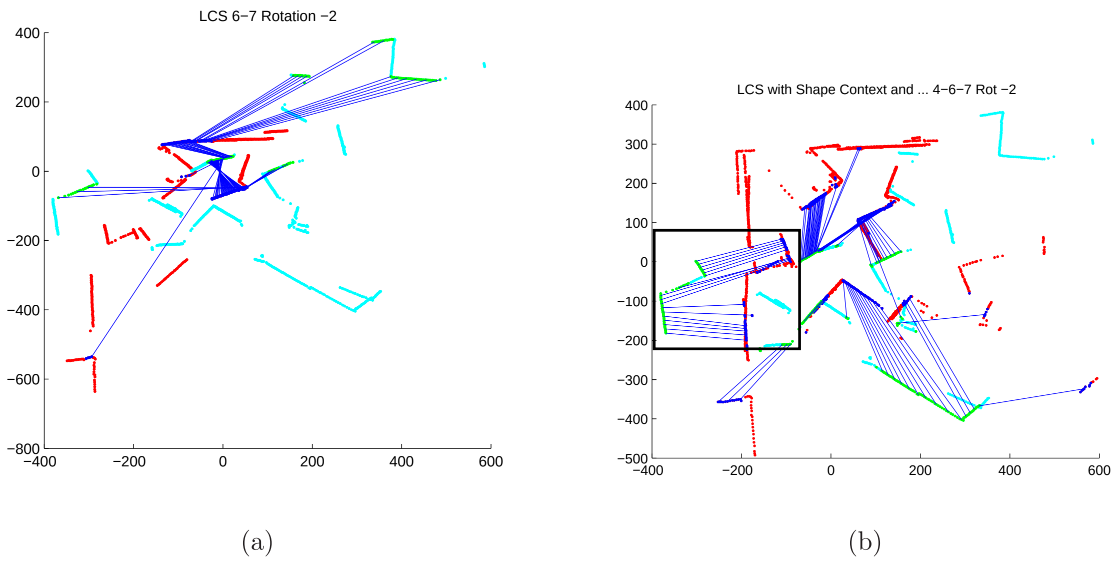


Figure 6.4: The corresponding points computed using shape similarity in (a) between scans 6 (red) and 7 (cyan) are all false positives. (b) The rectangular region shows the true positive correspondences between a partial global map composed of scans $4-5-6$ (red) and scan 7 (cyan). The blue lines link the corresponding points.

eliminate all other point correspondences that are farther than the average distance. We pick the translation that gives us the largest number of point correspondences. The main difference between segment filter and this step is that in segment filter we eliminate false positives segment wise, whereas in this case we do not use abstraction of segment for elimination - we eliminate all bad corresponding point pairs.

- **Scan overlap measure** As mentioned in Section 6.2.3 we perform the process for several rotations. For these experiments we rotated query scans from -2° to 2° with 1° interval. After we translate the query scan to target scan based on the best translation, we compute the number of points in query scan that are *close*, within a distance threshold to the points in the target scan. The distance threshold is computed dynamically as follows. We compute the average distance between correspondences in the segment that was picked by the segment wise translation process. The distance threshold is the average distance plus $40cm$. We consider only those point correspondences for which the tangent directions of the corresponding points are within 10° shift. The length of this correspondence is the scan overlap measure.

6.4 Global Map Construction

Currently the best method for mapping robot scans is using particle filters [251]. Particle filtering is a powerful tool for mapping robot scans but it has two major constraints. One is that there needs to be large overlaps between immediate scans and also that the environment is not very cluttered or very dynamic. The other is that the pose estimates are good. We compute very good pose estimates but since the overlaps between immediate scans are small and the environment is cluttered particle filters fail in our case as can be seen in Fig. 6.7(b).

Therefore, we use a deterministic technique to construct the global map and correct the errors in robot poses. We use *Iterative Closest Point (ICP)* [91]. We align maps by the translation and rotation obtained using our shape similarity approach. The correspondences used for ICP are selected using distance and tangent thresholds. Points from two scans should be below a distance threshold and also within a tangent threshold to correspond. The distance threshold is dynamically calculated using the average distance between the scan points aligned using pose estimates computed. We run the ICP algorithm iteratively as long as the quality of alignment increases with an an upper bound of 20 iterations. The quality is measured using scan overlap measure and the average distance among the corresponding points in the scan overlap (overlap distance). If in an iteration the scan overlap measure decreases and the overlap distance increases then we roll back the iteration and break the ICP loop.

6.5 Experimental Results

We use benchmark data from NIST. NIST collected robot scans in a simulated rescue arena at 16 different positions. The data we have is the laser range data without odometry. Four scans are taken at each position at four different orientations that differ by 90°. Each complete scan is composed of 1604 readings, 401 in each direction with the angle span of 100° and with 0.25° scan interval. Fig. 6.6 shows the estimated and actual robot poses and the error plot of the estimated robot poses. Our estimate is very good and we correct the error in estimates using ICP as discussed below.

6.5.1 Decoupled ICP

Since each scan is composed of four subscans taken after 90° rotations, we perform decoupled ICP after the ICP on complete scan. The measurements between different regions have overlaps and manually introduced errors due to the data acquisition process. The ICP applied to the complete scan cannot correct these local errors without sacrificing quality of global alignment. Hence we run the ICP algorithm on the four regions separately. We iterate on each region with quality check as explained in the Section 6.4. Fig. 6.7(c) shows the global map generated by our approach. For comparison, we show in (d) the map based on manually measured odometry. The global map generated using currently the best SLAM approach (based on particle filtering) is shown in (b). It was obtained using the CARMEN software described in Section 6.5.2.

6.5.2 CARMEN

CARMEN is a robot navigation software package developed by CMU. It has various modules useful for autonomous robots. The scan matching module of CARMEN is called VASCO and it is used to generate global maps from the robot scans. We input the data from NIST with the estimated odometry computed by the proposed approach to VASCO. Our input to VASCO is shown in Fig. 6.7(a). As can be seen in (b), VASCO was not able to improve the input map (shown in (a)).

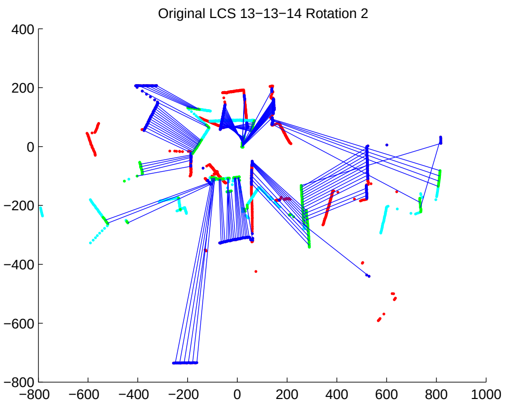
6.6 Conclusion and Future work

We have demonstrated that shape features of scans can be very powerful in scan matching in cluttered environment like rescue situations or rapidly changing environments. Our shape similarity technique is very useful in extracting overlaps among scans even when consecutive maps do not have large overlaps.

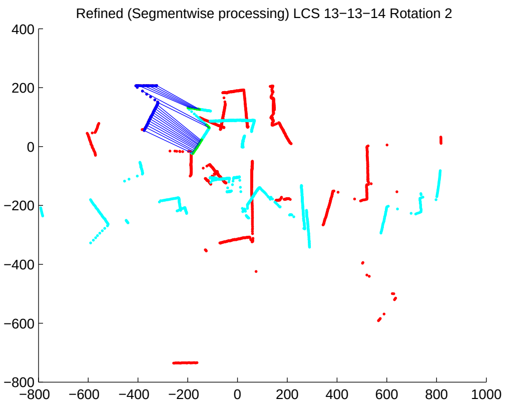
Once we get the overlaps using shape we can generate a prior distribution of the particles and use the particle filtering approach. We are working on using particle filters with the prior generated by our approach to compute the actual robot pose.

Acknowledgment

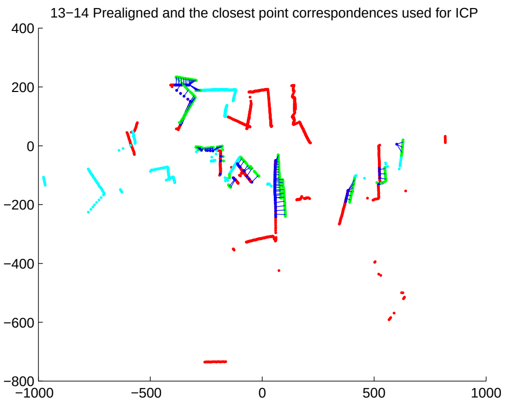
We would like to acknowledge the support from the National Institute of Standards and Technology, award Nr. NIST 70NANB5H11119, and discussions with Elena Messina and Adam Jacoff. We thank Dirk Haehnel for his prompt help in using CARMEN's scan matching module.



(a)

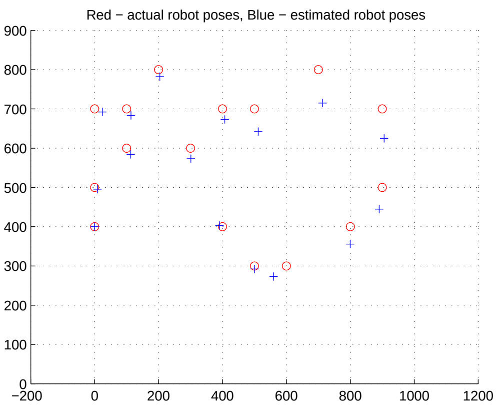


(b)

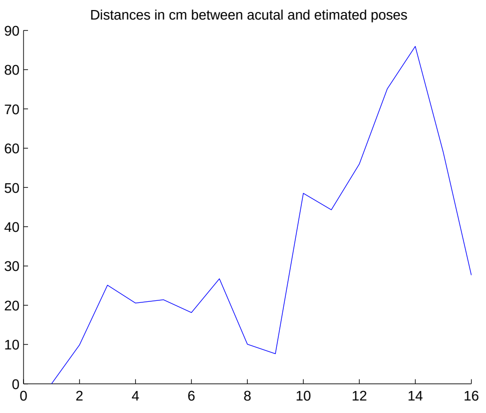


(c)

Figure 6.5: This figure shows the effect of the segment filter. (a) The corresponding points between scans 13 and 14 extracted using LCS. (b) Actual correspondence after passing through our filters. We can see that only 3 good segments are left. (c) 13^{th} and 14^{th} scans aligned using segment wise transformation described in Section 6.3.

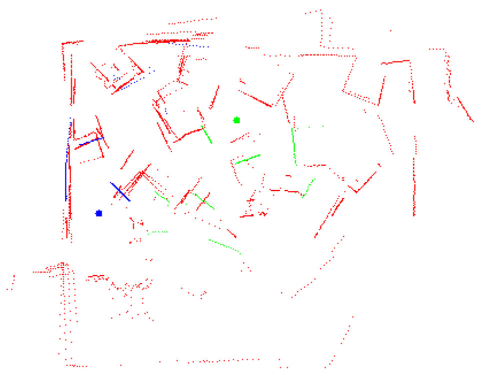


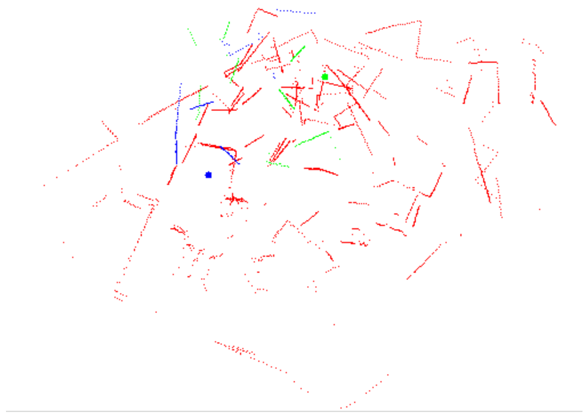
(a)

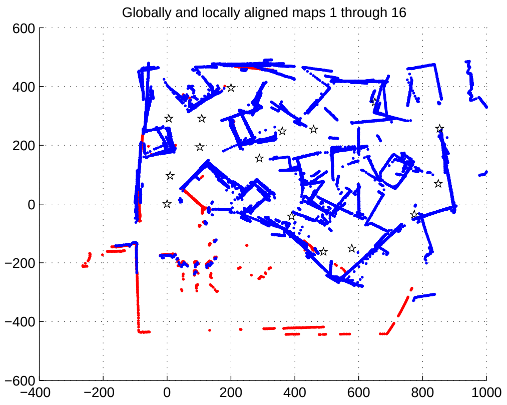


(b)

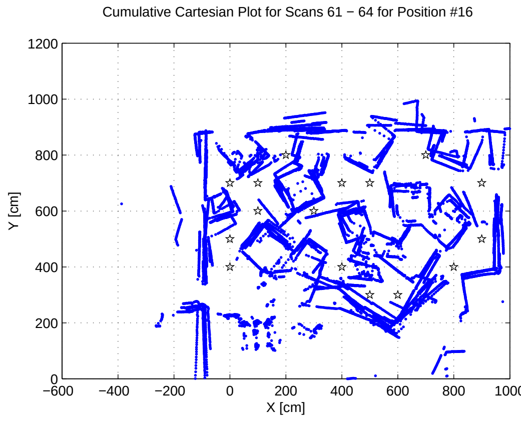
Figure 6.6: (a) Actual (plus) and estimated (circle) robot poses. (b) Error plot of the distance differences between the estimated and actual robot poses.


(a)


(b)



(c)



(d)

Figure 6.7: (a) The input global map of NIST log data with the odometry estimated by the proposed approach . (b) The global map generated by VASCO module of CARMEN. (c) The global map generated by our approach. (d) The global map of NIST log data.

CHAPTER 7

Force Field Simulation based Mapping

This chapter describes a novel approach, called Force Field Simulation to multi robot mapping that works under the constraints given in autonomous search and rescue robotics. Extremely poor pre-alignment, lack of landmarks and minimal overlap between scans are the main challenges. The presented algorithm solves the alignment problem of such laser scans utilizing a gradient descent approach motivated by physics, namely simulation of movement of masses in gravitational fields, but exchanges laws of physics with constraints given by human perception. Experiments on different real world data sets show the successful application of the algorithm. We also provide an experimental comparison with classical ICP implementation and a Lu/Milios-type alignment algorithm.

7.1 Introduction

This paper focuses on multi robot mapping in the field of Urban Search and Rescue Robotics ("rescue robots"). The task of multi robot mapping in rescue environments imposes especially challenging constraints:

- no precise or reliable odometry can be assumed, which means especially that the robots' relative poses are unknown
- due to the nature of catastrophe scenarios no distinct landmarks are given
- the overlap between pairs of the robots' scans is minimal

There are two reasons for our selection of rescue robots: (1) Rescue robots belong to the class of most advanced robotic systems. Their goal is to search a catastrophic site (like a collapsed building) for survivors. This requires building a map, and once victims are found, to localize them in the constructed map. First responders can then use this map to identify the safest way to rescue the victims. (2) The performance of rescue robots is systematically evaluated in the RoboCupRescue competition series with performance metrics developed by NIST [135]. The challenges involved in search and rescue applications provide objective evaluation of robotic implementations in representative environments, and promote collaboration between researchers. The robots must demonstrate capabilities in mobility, sensory perception, planning, mapping, and practical operator interfaces, while searching for simulated victims in a maze of increasingly difficult obstacles. The competitions are hosted on Reference Test Arenas for Urban Search and Rescue Robots, which were developed by NIST and were proliferated globally. The arenas provide a continuum of increasingly difficult representations of collapsed buildings that have simulated victims scattered throughout.

This paper introduces of a new process, called 'Force Field Simulation', shortly FFS, which is tailored to align maps under the aforementioned constraints. It is motivated by simulation of dynamics of rigid bodies in gravitational fields, but replaces laws of physics with constraints derived from human perception. It is an approach of the family of gradient descent algorithms, applied to find an optimal transformation of local maps (in particular, laser range scans) to build a global map based on feature correspondences between the local maps. Fig. 7.1 shows the basic principle: forces (red arrows) are computed between 4 single scans (the 4 corners). The scans are iteratively transformed by translation and rotation until a stable configuration is gained. The single scans are not merged, but kept separated. As they are moved according to the laws of the motion of rigid bodies in a force field, single scans are not deformed. FFS has the following properties:

1. Low level correspondences (data point correspondences) are not made by a hard decision (an integral of forces between pairs of points defines the force field in place of hard 'nearest neighbor' correspondences)
2. FFS is a gradient approach, it does not commit to an optimal solution in each iteration step
3. The iteration step towards an optimal solution is steered by a 'cooling process', that allows to jump the system out of local minima
4. FFS transforms all scans simultaneously
5. FFS easily incorporates structural similarity modeling human perception to emphasize/strengthen the correspondences

7.2 Related work

The problem of aligning n scans has been treated as estimating sets of poses [183]. Since sets of poses and the associated structures (maps) are conditionally independent, this estimation

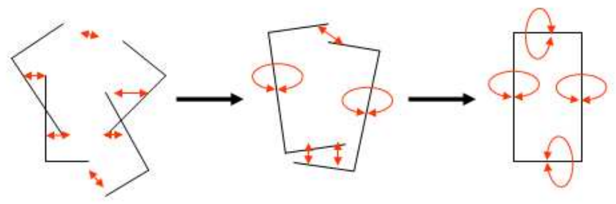


Figure 7.1: Basic principle of FFS. Forces are computed between 4 single scans. Red arrows illustrate the principle of forces. The scans are iteratively (here: 2 iterations) transformed by translation and rotation until a stable configuration is achieved.

is Simultaneous Localization and Mapping. The conditional independence is e.g. the key for Rao-Blackwellization (factoring the posterior of maps) of particle filters for SLAM [198].

There have been several algorithms to estimate the sets [206, 97, 98, 227, 193, 147]. The underlying framework for all such techniques is to optimize a constraint-graph, in which nodes are features, poses and edges are constraints built using various observations and measurements like odometry, scan-matching of range scans. These techniques differ in

- how they represent graphs - e.g. [97] uses a sophisticated data structure called Tree-map, [227] represents using sparse extended information filters (SEIF).
- how they build constraints - e.g. [183] uses linearized constraints obtained from scan-matching and odometry, [206] works with non-linear constraints.
- how they optimize the graphs - e.g. [206] uses stochastic gradient descent for approximate optima, borrowing the ideas from learning theory, [183] solves for exact optima using brute-force, [98] use Gauss-Seidel relaxation again for approximate optima.

All these approaches have performed well in many practical cases but they have one drawback that is they are sensitive to behavior of error models of sensors because of several assumptions and approximations which might not hold with sparse sensing.

[183] linearizes constraints by linearizing pose-relations, solving a linear equation of the form $AX = B$ to estimate X, the set of poses. This requires that A is invertible, so they conjecture that A is invertible if the constraint-graph is fully connected and the errors of the observations behave in a gaussian/normal way. [31] extends the same technique for 3D scans.

[206] presents an approximate optimization of non-linear constraints and demonstrate that their approach of approximating the optimization process in non-linear state space yields superior results compared to finding exact optima by approximating a non-linear state space (SLAM) to a linear state space.

Another strategy of attacking the problem is to treat the problem of SLAM from a perspective of aligning n scans simultaneously. The algorithms exploiting this perspective build from image registration techniques, the most famous being Iterative Closest Point (ICP) [20, 48] and it's numerous variants to improve speed and converge basins [226] and [185, 23]. Basically all these techniques do search in transformation space trying to find the set of pair-wise transformations of scans by optimizing some function defined on transformation space. The techniques vary in defining the optimization functions that range from being error metrics like "sum of least square distances" to quality metrics like "image distance" as in [24]. Their optimization process itself can be gradient descent or hill climbing or using genetic programming strategy as in [221]. All of these techniques have one major limitation, which is they search in *pair-wise* transformation space. Though in some variants of ICP the error from all pair-wise transformations is spread across all transformations to simultaneously align all scans, the procedure can be highly sensitive to outliers [225].

FFS also adapted the perspective of aligning n scans, it treats the alignment problem as an optimization problem. Rather than using a least squares solution to compute intermediate motions, FFS uses an iterative gradient technique to solve for (local) optima. Here FFS is similar to the approach proposed by [67], which simulates a dynamic spring system to register multiple range scans simulatenously. They describe the advantages of such a gradient descent system as follows: *'The reason [not to use a least square solution] is that the effects of any significantly incorrect correspondencens are compounded when the best alignment is computed (...) With a dynamic system it is possible to move in the direction of an intermediate solution without being totally committed to it'.* [67] differ in the choice of the registration function, which in contrast to FFS is based on one to one correspondences between points, as well as in the optimization technique.

FFS uses a gradient method with decreasing step width Δ_t. The registration function (target function) of FFS is based on Gaussian fields, similar to [33]. In contrast to [33], FFS uses a variable, decreasing σ for each iteration step t. Additionally, [33] solve the optimum of the registration using a quasi-Newton method, hence they do not steer the system with a step width parameter.

Since we keep the single scans separated, our search space is high dimensional, in the $2D$ case it is $3n$-dimensional ($3D$: $6n$-dimensional), with n being the number of scans. For example, our experiment described in section 7.4.2 uses 60 scans, our search space is therefore 180-dimensional. Birk and Carpin use a random walk technique to reach the optimal solution. Since random walk techniques tend to become critical in high dimensions, we do not utilize this technique in our approach but decide in favor of a guided (gradient) walk.

This search in high dimensional space at first sight seems very complicated, demanding computation of high dimensional gradient but fortunately using potential field simulation for various computer vision tasks like contour detection, segmentation, registration has been empirically successful [278, 136, 277], [13], [204], [263]. Since mapping is closely related to registration, the approaches whose motivations are closely related to our approach are [22, 13, 67]. In [67] they align range scans by moving them simultaneously. The movements are not just based on the minimizing error of transformation computed using correspondences but on the simulated fields generated by imaginary springs attached to the corresponding points. Our technique differs from [67] in that the force field is generated not just by closest point correspondences but using perceptual principles and gaussian fields similar to [33]. [22] also performs search in $3n$ dimensional space. For each configuration they compute energy as the sum of the Normal Distribution Transforms (NDT) [21] of all the scans in the configuration and update the configuration using Newton's optimization algorithm that involves the first and second derivatives of the energy. Their approach is very closely related to ours but does not use perceptual features and rigid body dynamics and hence in principle can be more sensitive to outliers.

7.3 Force field based mapping

The following motivates and describes the FFS algorithm. A pseudo code representation can be found at the end of this section.

7.3.1 Basic Principle

To draw the analogy to Newtonian Physics, each scan s_i can be seen as a rigid body of masses: the scan points represent the masses, rigidly connected by massless rods. A global map g defines the transformation of all scans, it therewith defines the distribution of all masses (the union of all scan points). In the framework of Newtonian Physics the gravitational forces between these masses forms a gradient field. The FFS algorithm is motivated by simulation of the movement of bodies in a gradient field. In contrast to pure physics it replaces physical principles of masses and forces by principles that correspond to human visual perception, i.e. gravitation is replaced by 'strength of correspondence'. Also, to achieve convergence to a stable state of minimal total energy, the kinetic energy is not taken into account, i.e. the velocity of each rigid body after each iteration step is set to 0. Also FFS uses a 'cooling' strategy in its step width parameter that initially adds energy to the system to allow for escape from local minima, see section 7.3.2.

Let $\mathcal{S} = s_1, \ldots, s_n$ be a set of n scans gained from laser range scan devices. A scan $s_i = (p_1^i, \ldots, p_j^i)$ consists of j data points. Data points are the coordinates of reflection points of the laser range scanner in a local coordinate frame defined by a single robot pose. We also assign a scalar value, a *mass* m_j^i to each data point, which can be interpreted as the perceptual importance. For the purpose of multi robot mapping we assume that each scan is possibly gained from a different robot, while the robots' relative poses are unknown or poorly estimated. The task of the algorithm is an optimization over the set of the robots' poses; hence the goal is to find transformations for all n scans $s_{i=1..n}$ to registrate the scans, such that similar features in different scans match 'perceptually consistently' when they are superimposed on top of each other.

Observe that the order of local maps is irrelevant in our framework since we transform the scans simultaneously, which is an important property of the algorithm to be applicable for multi robot mapping. For single robot mapping, FFS is canonically extandable to online FFS, see section 7.3.6.

The transformations performed are rotation θ_i and translation x_i, y_i of each scan s_i. Superimposing the transformed maps builds a *global map* g as shown in Fig. 7.2. During the global

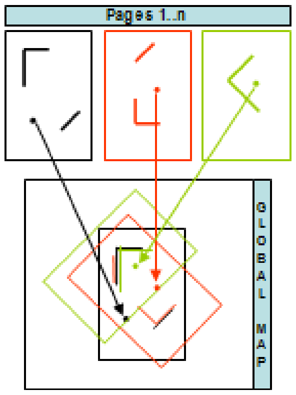


Figure 7.2: Single scans are transformed to build a global map

map building process single maps are not merged but kept separated. Therefore a global map is defined by the vector of the transformation parameters of all scans: a *global map* $g = (t_1, \ldots, t_n)$ is a $3n$-dimensional vector of transformation parameters $t_i = (x_i, y_i, \theta_i)$ of all n scans $s_{1\ldots n}$. The space of all global maps is denoted by $\mathcal{G}$. To registrate the scans,we define a *fitness measure* P_g to evaluate the 'perceptual consistency' of a global map g. Finding the global map g_k that minimizes P_g is clearly an optimization problem. FFS solves this optimization problem with a gradient technique that iteratively transforms all scans simultaneously until a stable configuration (local minimum) is reached. The following section will motivate and define the fitness measure P_g as well as the implementation of the gradient approach.

Correspondence function

As in [33], the basic idea of our registration method is to use a Gaussian field to define a strength of correspondence between data points, i.e. a measure for both spatial proximity and visual similarity of two points belonging to different scans.

A *correspondence* between data point p_1 and a data point p_2 is defined as a vector

$$V(p_1, p_2) = C(p_1, p_2) \frac{p_2 - p_1}{\|p_2 - p_1\|} \tag{7.1}$$

Its magnitude $\|V(p_1, p_2)\| = C(p_1, p_2)$ describes the strength of correspondence, defined as:

$$C(p_1, p_2) = \frac{1}{\sigma_t \sqrt{2\pi}} e^{\left(-\frac{\|p_2 - p_1\|^2}{2\sigma_t^2}\right)} m_1 m_2 \cos(\angle(p_1, p_2)) \tag{7.2}$$

with m_i being the mass assigned to p_i (see section 7.3.4), and the angle $\angle(p_1, p_2)$ being the angle between the *directions of points* p_1, p_2, which will be defined in section 7.3.3. Intuitively, the direction of a point is the direction of an underlying model of a linear structure (a line segment). Expression 7.2 can be interpreted as a force field whose sources are located in the data points. It has the following properties:

1. The strength of correspondence decays with Euclidean distance, the influence of distance is controlled by the parameter σ_t

2. The strength of correspondence is weighted by the mass of each data point and depends on the angle between point directions, i.e. it is 0 for orthogonal directions, 1 for parallel directions.

We propose this model for the following reasons:

1. Distance likelihood and parallelism follow the basic principles of Gestalt Psychology [270], modeling low level cognition.

2. The scale parameter σ_t gives additional freedom to adjust the process (see section 7.3.2), it enables the correspondence process to work on different visual scales.

3. Assigning a mass to a data point can be seen as assigning a visual importance to it. Data points in regions of interest, computed by mid level cognitive modules, will be assigned higher masses. See section 7.3.4 for further details.

In terms of the physical framework, the correspondence $V(p_1, p_2)$ (eq. 7.1) describes a force on p_1 towards p_2 with strength $C(p_1, p_2)$. Embedding the scans s_i into $\mathbb{R}^2$ using the transformations defined by a global map g, we can define a vector field $F : \mathbb{R}^2 \supset \mathcal{P} \to \mathbb{R}^2$, the *Force Field* on the set of all points $\mathcal{P} = \{p | p \in \bigcup_{i=1..n} s_i\}$ by summing the correspondences.

$$F(p_i) = \sum_{p_j \in \mathcal{P} \setminus p_i} V(p_i, p_j) \tag{7.3}$$

By definition of the strength of correspondence F is radial and hence a gradient field. With

$$A = m_1 m_2 \cos(\angle(p_1, p_2))$$

the overlying potential is defined by

$$P(p_i) = \frac{1}{2} \sum_{p_j \in \mathcal{P} \setminus p_i} \int_{\infty}^{r} \frac{A}{\sigma_t \sqrt{2\pi}} e^{\left(-\frac{z^2}{2\sigma_t^2}\right)} dz \tag{7.4}$$

with $r = \sqrt{(X - x)^2 + (Y - y)^2}$,$p_i = (X, Y)$,$p_j = (x, y) \in \mathcal{P}$.

Note: $P(p_i)$ is the potential over F since:

$$F(p_i) = - \bigtriangledown P$$

$$= \begin{bmatrix} -\frac{\partial \mathcal{P}}{\partial x} \\ \\ -\frac{\partial \mathcal{P}}{\partial y} \end{bmatrix} = \begin{bmatrix} \sum_{p_j \in \mathcal{P} \setminus p_i} \frac{A}{\sigma_t \sqrt{2\pi}} e^{\frac{-r^2}{2\sigma_t^2}} . \frac{X-x}{r} \\ \\ \sum_{p_j \in \mathcal{P} \setminus p_i} \frac{A}{\sigma_t \sqrt{2\pi}} e^{\frac{-r^2}{2\sigma_t^2}} . \frac{Y-y}{r} \end{bmatrix} = \sum_{p_j \in \mathcal{P} \setminus p_i} \frac{A}{\sigma_t \sqrt{2\pi}} e^{\frac{-r^2}{2\sigma_t^2}} . \vec{u} = \sum_{p_j \in \mathcal{P} \setminus p_i} V(p_i, p_j)$$

where $\vec{u} = \frac{p_i - p_j}{\|p_i - p_j\|}$.

Finally, we define the *fitness measure* or *potential of a global map* $g \in \mathcal{G}$ as the sum of potentials of all data points $p \in (P)$:

$$P(g) = \sum_{p_i \in \mathcal{P}} P(p_i) \tag{7.5}$$

In this framework, the potential $P(g)$ can be interpreted as the weighted average distance of all point pairs of different scans, the weight being the strength of correspondence. This potential can be seen as the quality of registration achieved by the transformations defined by g. To minimize the potential we apply an iterative gradient descent approach, the gradients in each data point given by $F(p_i)$, eq. 7.3. Computing the correspondences explicitly gives us these gradients , hence in the implementation of the algorithm there's no need to explicitly compute and derive the potential $P(g)$ (eq. 7.5)for the actual gradient descent.

In FFS, the computation of the transformation of each scan is determined assuming movement of rigid bodies in the given gradient field, i.e. all data points $p_i^j \subset \mathcal{P}$ of a single scan s_j share the same transformation, consisting of rotation and translation. However, eq. 7.3 does assume a non-rigid, independent movement of the data points, also the points' potential $P(p_i)$ (eq. 7.4) is defined over the space of all single (not rigidly connected) point configurations, which is a $2m$ dimensional space with $m = |\mathcal{P}|$. This means, the gradient $F(p_i)$ is defined under the assumption of unrestricted freedom of movement in $\mathbb{R}^2$. To implement rigid body movement, we have to impose a restriction on the movement. The restriction is defined by the possible point configurations that are allowed by the transformations $g \in \mathcal{G}$. The laws of rigid body dynamics define these constraints: computing the gradient $F(p_i)$ in each data point as in eq. 7.3 results in a $2m$-dimensional gradient vector, the laws of rigid body dynamics map this vector to a $3n$ dimensional vector $\bigtriangledown g$. $\bigtriangledown g$ describes the transformation of all n scans such that each data point moves in the direction of maximum descent of $P(g)$ in eq. 7.5 , i.e. under the rigid body constraints. We therefore achieve a movement in the direction of the gradient gained in the $2m$ dimensional single point space projected onto the restricted $3n$ dimensional rigid movement subspace.

The basic laws of dynamics of rigid bodies in force fields accumulate the translation of all masses of a single scan into a single translation and a single rotation around a defined center. For each scan s_i, the translational and rotational acceleration has to be determined. The translational acceleration $a_T(s_i)$ of a scan s_i is defined by:

$$a_T(s_i) = \frac{\sum_{p \in s_i} F(p)}{\sum_{p \in s_i} m_p} \tag{7.6}$$

The rotational acceleration a_R is computed by torque and moment of inertia. Torque and inertia play the role of force and mass respectively, but take into account the distance to the rotational center c_R.

$$\begin{aligned} inertia &= \sum_{p \in s_i} m_i \|p_i - c_R\|^2 \\ torque &= \sum_{p \in s_i} \|p_i - c_R\| \times F(p) \end{aligned}$$

a_R is defined as

$$a_R = \frac{torque}{inertia} \tag{7.7}$$

The rotational center c_R is either defined as the robot's position, or by the center of mass. Experiments show, that in the first iteration steps it is useful to set the rotational center to the center of mass, while in later steps the robot's position is preferable. The first choice enables easier rotation, the second is modeling the actual scan setting more precisely. Hence, the closer the global map is to the solution, the more preferable is the robot's position as rotational center.

With a_T and a_R the transformation $t_k = (x_k, y_k, \theta_k)$ for scan s_k is defined by:

$$(x_k, y_k) = \frac{1}{2} a_T \Delta_t^2 \tag{7.8}$$

$$\theta_k = \frac{1}{2} a_R \Delta_t^2 \tag{7.9}$$

Δ_t being the step width of the gradient descent, as described in section 7.3.2.

With these constraints, the gradient in each iteration is computed by the following steps:

1. for each pair of points $p_i, p_j \in \mathcal{P}$ compute $V(p_i, p_j)$
2. for each point $p_i \in \mathcal{P}$ compute $F(p_i)$
3. for each scan $s_k \in \mathcal{S}$ compute the transformation $t_k = (x_k, y_k, \theta_k)$ using the points $p_i^k \in s_k$. This step results in a $3n$ dimensional gradient vector $\bigtriangledown g$

Computing all correspondences V in step one is an $O(n^2)$ process, section 7.3.5 will deal with the necessary reduction of computational complexity.

Fig. 7.3 shows two iteration steps of FFS using two simple scans, consisting of three and five data points. In the left figure, the forces $F(p_i)$ in each data point p_i are plotted as green dotted lines.The two scans are transformed until they are superimposed, i.e. a stable configuration (local minimum of $P(g)$) is reached. As in all gradient descent methods, the determination of the step width Δ_t is crucial. Also, gradient methods imply the danger of being trapped in local minima. We tackle both problems with the determination of step with Δ_t and σ_t as described in the following section.

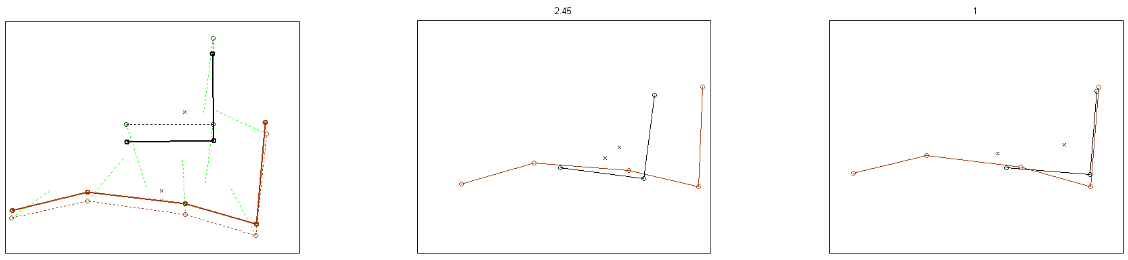


Figure 7.3: Left: Two scans (black/brown) superimposed. Dotted lines: scans at time t, green lines: forces on the data points at time t. Solid lines: scans after one iteration, time $t + \Delta t$. Both scans are translated/rotated according to the forces. Center: after iteration 5. Right: iteration 10.

7.3.2 Cooling down the motion: time stepping Δ_t and parameter σ_t

The determination of step width parameter Δ_t in any gradient descent approach is a well known problem. Δ_t chosen too small results in inapplicably slow convergence behavior and is not robust too noise, Δ_t chosen too big might miss the optimum. In FFS, the step width Δ_t is used as a steering parameter of the algorithm in connection with the parameter σ_t, which determines the influence of distance in the correspondence function. We designed Δ_t as exponentially decreasing, σ_t linearly decreasing.

A large Δ_t allows the scans to be massively relocated (shuffled), they overshoot their correct position in the direction of the correspondence gradient. Naturally, a small Δ_t moves the scans less (the amount of replacement is directly proportional to Δ_t^2, as defined by the laws of movement). We chose the strategy of decreasing Δ_t and σ_t experimentally, having analogies of the cooling behavior of algorithms like simulated annealing in mind. The imprecise, non optimal large Δ_t at the beginning allows the system to possibly escape from local minima. Observe that in contrast to a technique like simulated annealing we cool down a gradient guided process, not a random state change or a random walk technique that would not be applicable in our high dimensional search space. We therefore avoid the problems with a high computational load (high number of iteration steps) that tend to appear in simulated annealing due to unguided selection of the next state.

The parameter σ_t in eq. 7.2 steers the influence of distance in the computation of point correspondences . A large σ_t enhances the relative influence of data correspondences with greater distances, and, since it equalizes this spatial proximity property, favors the influence of visual similarity. A small σ_t emphasizes local proximity, which is useful if the global map is already close to an optimum.

The effect of cooling is shown in fig. 7.10 showing our experimental results on the 'apartment data set'. Observe that the potential function (the fitness measure) as shown in fig. 7.11 is not monotonically decreasing in the first iteration steps. This shows an 'overshooting' of the system due to a large Δ_t, in the data registration this can indicate an escape from a local minimum.

It is important to mention that after each iteration the system resets the velocity of each scan to zero. This guarantees that the system converges to a stable state (assuming $\Delta_t \to 0$).

7.3.3 Point direction and optional resampling

The 'point direction' is used in the correspondence eq. 7.2 to assign to points the direction of an underlying linear structure. It is derived by modeling the point set with line segments using the extended EM algorithm described in [168]. Utilizing a segment split and merge approach, the extended EM algorithm automatically adjusts number and location of the line segments in a way such that linear structures are represented (by a single, possibly long line segment) as well as round structures (by multiple short segments). Hence, even scenarios not being rich in linear structures are robustly represented. The algorithm was already successfully applied to model indoor and outdoor rescue scenarios. A $3D$ version of this algorithm for approximation of scan points with planar patches is described in [162].

The data of each scan s_i is approximated by a set of line segments L_i. The direction of a point p in s_i is the angular direction (in the scan's coordinate frame) of the closest line segment in L_i. The approximation of the data set with line segments results in a very stable and intuitive estimation of point directions. Fig. 7.4 shows the influence of point directions for the correspondences. The

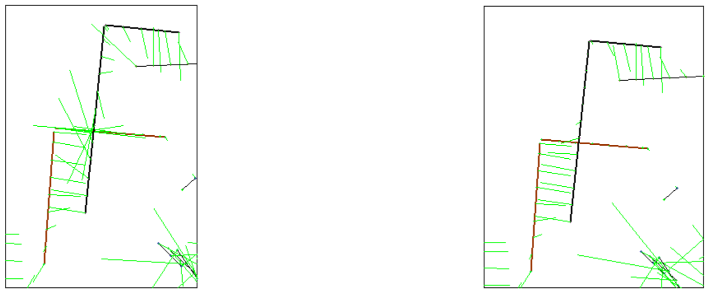


Figure 7.4: Left: forces between two scans (red lines belong to first scan, black lines to second scan) computed without direction information. Right: forces computed using L and K as described in example 2 (using equal masses). Correspondences between non parallel structures are weakened.

closest line segment l_j to the point p is called the *supporting line segment* if its distance to p is below a certain threshold. Points without supporting line segments are removed from the data set. Due to the nature of the extended EM algorithm, these removed points are points in areas with low point density. Low point density results from objects which are hit less than others. This is the result of either erroneous scanning, non static objects, or low scanning density, which by itself results from either long distance to an object or simply the fact that a certain location was only scanned a few times. All of these topics include uncertainty about the existence of the object, hence we disregard such points. The behavior of the extended EM segment fitting guarantees that safe, distinct objects are not removed. The removal of uncertain data increases the stability of the FFS algorithm.

Having the segments, the data can optionally be resampled along the supported line segments with an equal sampling distance. Such a point set has a more homogeneous distribution of points, which tends to be advantageous: experiments showed that homogeneous distributions are helpful to avoid local minima if the configuration of scans is still far from the optimal solution, since over represented areas (e.g. features with unusual high scanning density due to multiple scans in a single location) are equalized. Additionally, the optional resampling can significantly speed up the computation if the number of data points reduces drastically due to the chosen sampling resolution, see section 7.3.5. If the data is resampled, only the line segments (two endpoints) are stored, also resulting in a significant data compression (typically about 1 : 100).

7.3.4 Regions of interest

A major difference to the pure physics simulation is that the mass values assigned to the data points are not assumed to be constant. The mass m_i for a point p_i is used to compute the force as in eq. 7.3, yet it can be reassigned a different value for the computation of movement of the scan (we are not modeling physics but perception, hence freedom from Newton's law is given). Steering the mass enables the algorithm to react better to perceptual properties: there is no perceptual reason for an 'important point', e.g. a corner point, assigned a high mass for force computation, to be less mobile than other points during movement computation (caused by its high mass). This observation suggests using different masses during the computation of forces than during the computation of the movement.

To compute the mass distribution, we focus on cognitively interesting features in the global map by defining Regions of Interest, ROI. ROI are regions around certain features detected in the global map. Since the global map is analyzed, not single scans, features are detected even if they are not present in single scans but emerge from the overlap of scans, a case that is very likely in our assumed setting of sparse scans with little overlap (e.g. a corner that consists of one segment of scan 1 and one segment of scan 2 will be detected, although it might not be present in either scan). In general, once an interesting feature is detected, the ROI is defined as a region around the feature. To let FFS focus on these regions, the correspondences (forces) in these regions are emphasized. The emphasis is gained by the assignment of mass values: data points in ROI are assigned a higher mass during the force computation. Hence forces in regions of interest are stronger, i.e. the correspondences and therefore the rigid body movement are based on forces with focus on perceptually interesting features. There are many possible ROIs, in particular higher level objects as described in the spatial semantic hierarchy [157] could be used, e.g., corridor junctions represented with line segments. In the current implementation, ROI are defined by automatic corner detection as an example for ROIs. Though such an implementation restricts the applicability to the presence of corners, it can be easily exchanged to focus on different features. In this implementation we benefit again from the supporting line segments, as described in section 7.3.3: corners are gained from intersections between supporting line segments. Each intersection between two supporting line segments (extended to infinite lines) in the global map with a distance below a certain threshold to both segments and an intersection angle between 80° and 100° counts as a corner. The ROI are the union of discs with a given radius centered at all corners, see fig. 7.5(a). Data points in these regions will be assigned higher mass values (Fig. 7.5(a) does not show the data points but only the supporting lines). Fig. 7.5(b) shows a global map with ROI at an early iteration stage. Since the scans are still poorly arranged, the dislocation of superimposed scans leads to detection of many corners. However, long walls, represented by parallel lines (in fig. 7.5(b) the vertical lines on the left side, marked by arrows) are not emphasized. This allows for easier relocation in these 'wall areas' in favor of corner correspondences. Without ROI, point correspondences between data points of the wall areas (then equal in strength to correspondences between corners) would fix the scans' positions in a local minimum, not giving the necessary freedom to be relocated (intuitively: to slide along each other) with regard to the perceptually more important correspondences. In early iteration stages, the ROI are still wide spread around the correct feature positions, as seen in fig. 7.5(b). Since in later iteration steps the scans become better aligned, ROI become better focused and mark the positions around features in higher precision.

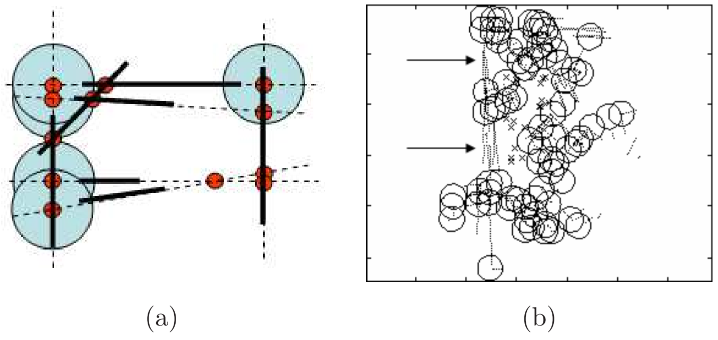


Figure 7.5: Regions of Interest (ROI). (a) line segments (black) and intersection points (red). Intersection points satisfying the constraints on angle of intersection and distance to segments define ROI (blue discs). (b) ROI of an early global map (upper left corner of the NIST data set (fig. 7.7). Arrows mark areas of lower interest ('wall-areas'), see text.

7.3.5 Computational complexity

Time complexity

The definition of C (eq. 7.2) on pairs of data points leads to an algorithm with $O(n^2)$ time complexity where n is the number of points. This is certainly prohibitive for real applications. Different techniques can be used to reduce the complexity by taking advantage of two main properties of equation 7.2:

1. For each data point only its local neighborhood must be examined, since the forces between points rapidly decrease with distance. Hence some techniques successfully built into ICP implementations (which suffers from the same complexity problem) can be used to reduce the complexity, e.g. KD-trees. In the current implementation, we take advantage of the line segment representation of the data; we use a bounding box intersection approach on axis aligned bounding boxes around the line segments: bounding boxes around all line segments are computed and extended by $2\sigma_t$ in each direction. The force between two data points is computed only if the two corresponding lines' bounding boxes overlap. Hence we first reach a computational complexity based on the number $m << n$ of line segments, which is significantly lower than the number n of data points. Secondly, though bounding box intersection is an O(m log m) computation (note again: m=number of segments), update techniques as reported in [51], reduce the expected complexity to $O(m)$. This linear complexity is reported under the constraint of 'relatively small' movements of objects, such that the $O(m \log m)$ sorting in the sweep and prune step reduces to $O(m)$ on a nearly presorted list. The constraint of small movements is met for most of the iteration steps in FFS. To give an idea of the order of magnitude of reduction that is achieved some numbers for the NIST data set (see section 7.4.2) should be mentioned:

   - 60 scans contain a total of 21420 points, represented by 332 line segments (on average: 65 points per segment)
   - average number of colliding pairs of segments per iteration 1500, hence we have $65 \times 65 \times 1500 (= 6,337,500)$ computations, compared to $21420^2 (460,000,000)$.

2. The data points have to be evaluated with a certain accuracy only. By approximating the evaluation of force field we can achieve computational reduction in the following two manners:

   - The current FFS implementation sub samples each segment equally with some sampling distance. For the NIST data set (sampling distance $10cm$), we achieve in average 7 data points per segment, the force computation is therefore reduced to $7 \times 7 \times 1500 (= 73,500)$ computations.
   - Greengard and Strain introduced Fast Gauss Transform (FGT) [112] which is in turn based on Fast Multipole Methods introduced for high speed simulation of particle dy-

namics in potential fields [111]. The main advantage with FGT is that the force field can be computed in linear time with a constant factor depending on the precision required in computation of the field. Details can be found in [112]. The main idea is to compute the force field using a divide and conquer strategy and exploiting Hermite and Taylor expansions. FGT has been first introduced in [68, 13].

Space complexity

Since we approximate the scans by segments , it is not necessary to keep the original data. For each scan, only the segments' endpoints have to be stored. Experiments with the extended EM algorithm [168] on $2D$ Laser data sets show an average compression rate of 1 : 100 (200 data points per segment).

7.3.6 Online FFS

The described algorithm easily can be extended to online SLAM, i.e. scans are recorded and processed subsequently, as they arrive from the laser device. The extension is canonical: each additional scan is pre aligned, then FFS runs on the previously aligned data set plus the new scan.The current FFS system targets the application of multi robot mapping, hence the sequential processing is not implemented yet.

Algorithm 1 Force field based mapping

```
 1: Compute supporting line segments (section 7.3.3)
 2: Resample data set(section 7.3.3)
 3: $S \leftarrow$ initial state of transformations
 4: initialize step width $\Delta_t$ and $\sigma_t$(section 7.3.2)
 5: repeat
 6:     Compute regions of interest (ROI) on global map (section 7.3.4)
 7:     Assign masses to data points with respect to ROI
 8:     Compute forces using $\sigma_t$ (eq. 7.3)
 9:     Assign constant mass values to data points
10:     Compute rotational and translational acceleration (eq. 7.7 and eq. 7.6)
11:     Compute transformations $\bigtriangledown g$
12:     $g \leftarrow g + \bigtriangledown g$
13:     Set velocity of all points to zero
14:     update $\sigma_t$ and $\Delta_t$ (section 7.3.2)
15: until average of relocations $>$ Threshold
```

7.3.7 Pre Alignment

The pre alignment does not make use of odometric sensor data, but is based on shape similarity. It finds distinct shape features in single scans and tries to find an optimal overlap based on the shape similarity of these features. For further details see [4].

7.4 Experimental results

7.4.1 Performance Comparison to classical ICP

Fig. 7.6 shows the difference between the results of aligning a hypothetical set of 3 simple scans using classical ICP and our approach. Due to the hard constraints of using the nearest point correspondence only ICP (top row) ends in a non perceptually optimal configuration.

FFS takes into account the correspondences between all points first, a decreasing σ_t finally guarantees the correct positioning of the scans, decreasing the influence of points being too far away. The bottom right image shows the result of FFS after 12 iteration steps.

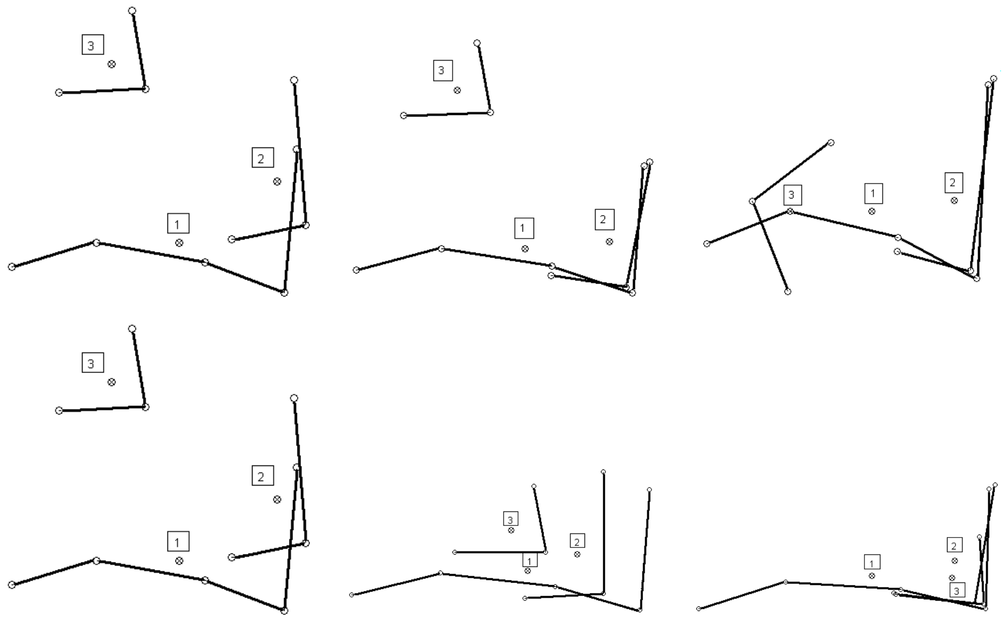


Figure 7.6: The top row shows 3 steps of the alignment of 3 scans (each scan consists of a single corner only) using classical ICP, the bottom-row shows the results of the proposed approach. The alignment progress can be seen from left to right in both cases. The square boxes show the robot poses of the the scans.

7.4.2 NIST disaster area

The NIST data set used in this experiment simulates a typical data set of multi robot mapping in rescue scenarios. It is especially complicated, as it matches the complicated constraints imposed by these settings, which contain only imprecise odometry, no landmarks and very little overlap.

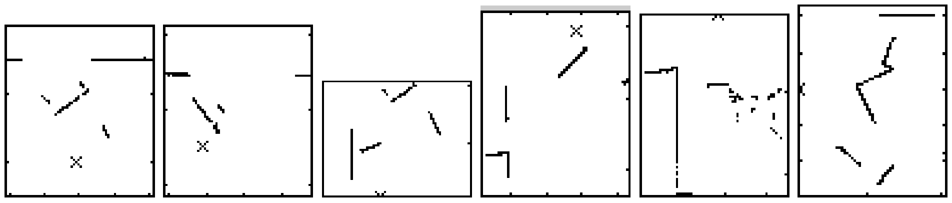


Figure 7.7: 6 out of 60 scans of the NIST rescue scenario data set. The scans in this data set are very sparse and have minimal overlap.

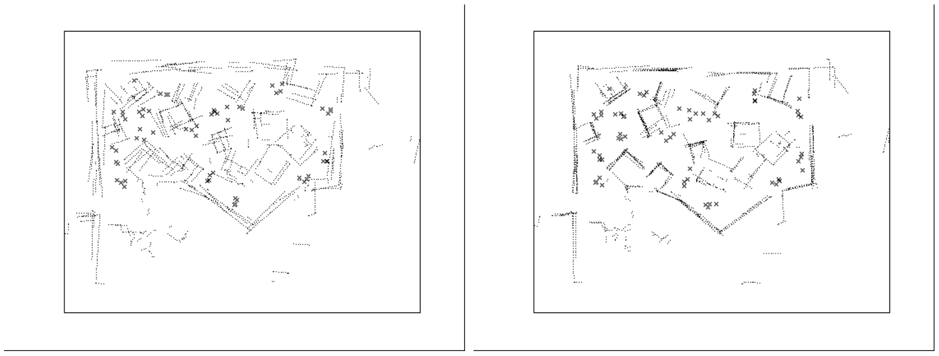


Figure 7.8: Left: 60 scans superimposed building a global map using a rough initial transformation estimation. Right: after 20 iterations of FFS. The crosses show the robots' positions. (the reader might try to find the 6 single scans of fig. 7.7 in the global map. $1, 2, 3$ and 6 are part of the upper left corner, 4 and 5 are located in the lower left corner). The final result of FFS is shown in fig. 7.9, left

The data set consists of 60 scans taken from 15 different positions in directions E,N,W,S with an overlap of $5°$ (i.e. overlap between E and N, N and W etc.). The area has a size of approx. $10 \times 15m$, the 15 locations differ approx. $2m$ from each other (see Fig. 7.7 for sample scans). The distance between the positions of the 4 scans taken from an assumed single position differs up to $30cm$, with an angular error of up to $20°$ to the assumed direction. The size of the arena was approx. $12 \times 10m$. This data set can be seen as a multi robot mapping scenario using 15 robots, with 4 scans gained from each robot. Although the pre-alignment assumes this setting, FFS actually

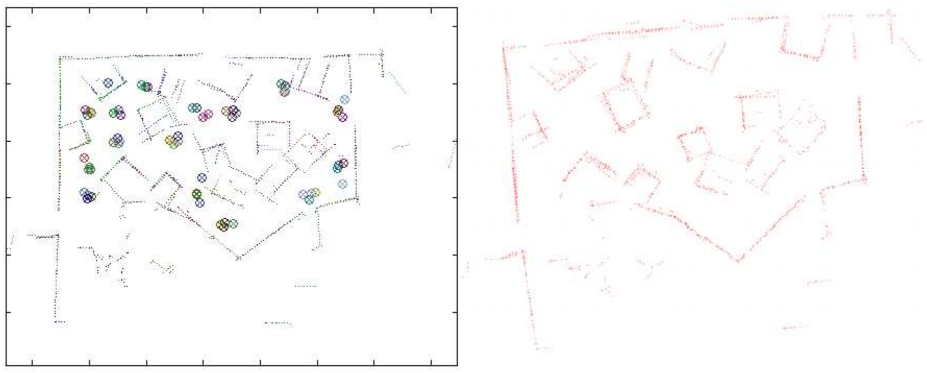


Figure 7.9: Left: final map (after 50 iterations) of intialization fig. 7.8 with FFS. Right: the final map obtained by the Lu & Milios technique as reported in [31]. The systems lead to results of comparable quality.

treats the 60 scans as independent scans without help of any further information, e.g. constraints on the groups of 4. The single scans have very little pair wise overlap. Fig. 7.7, shows 6 example scans, all located on the left side of the global map; the overlapping pairs among these scans are $(1,2),(1,3),(1,4),(2,3),(3,4),(3,6),(4,5)$.

The test performed on this complicated data set demonstrates the robustness of the FFS system. The initial, pre-alignment map is gained by the shape-based algorithm described in section 7.3.7. Fig. 7.8 shows the initial global map as well as iteration step 20, fig. 7.9 the final global map, after 50 iterations. The data set was re-sampled as described in section 7.3.3. The radius of the ROI was set to $5cm$, the parameters for the motion cooling were set as:

- Δ_t decreases from 5 to 1 with step factor of 0.96
- σ_t decreases from 15 to 4 with step size of 0.25

Although the data is poorly pre aligned and the overlap between the single scans is minimal, FFS successfully reconstructs the global map, which proves its applicability for this multi robot setting. The mean translation/rotation of the scans (translation/rotation between initial and final global map) is $16cm/4°$, the maximum translation/rotation is $50cm/10.5°$.

The alignment can also be seen as a movie at: `http://knight.cis.temple.edu/~lakaemper/FFS/FFSTheMovie.wmv`

The movie especially makes clear the effect of the motion cooling.

Fig. 7.9 shows the result after 50 iteration steps. The computational time for each iteration is 1 second on a 1.5GHz desktop computer in the current MATLAB implementation, using the bounding box approach as described in section 7.3.5, and resampling with a distance of $10cm$.

The sensitivity to initial conditions was tested as follows: using the result of the previously described experiment (fig. 7.9, left), we distorted this map by random transformations of the single scans in the range of $\pm 30cm$ (translation) and $\pm 20°$ (rotation). The visual appearance of such maps is similar to the appearance of the map shown in fig. 7.8, left. Taking these distorted maps as initial configurations, FFS achieved results that differed from the source map by a maximum of $0.9cm$ and $0.7°$, i.e. they were visually identical.

In order to compare the proposed FFS approach to the state of the art of existing robot mapping approaches, we applied three influential approaches to the NIST data set illustrated in fig. 9. We applied the particle filter based DP-SLAM [71], the ICP based VASCO robot mapping module of CARMEN, and improved grid-based SLAM with Rao-Blackwellized Particle Filters [116]. All three approaches failed to produce any reasonable results, since they are based on sequential processing of data (online SLAM), which can not be applied on this data set due to the extremely minimal overlap of consecutive scans (even if the order is known).

However, we compared to a recent implementation of Lu/Milios type SLAM [31]. The results are shown in fig. 7.9. Both algorithms show an overall comparable performance, although local differences can be seen: the Lu/Milios type SLAM reconstructs the top right corner better, while FFS performs better on the left side.

Fig. 7.11, left, shows the potential $P(g)$ vs. iteration curve for this data set. The potential is monotonically decreasing, hence in this case FFS steers directly towards a (local) minimum, which is reasonable due to the initialization. The next experiment will show a different case.

7.4.3 Apartment

This experiment demonstrates the benefits and applicability of FFS in data sets which are incorrectly pre aligned e.g. due to effects of wrong loop closing. We used the IROS 2006 test data set taken from `http://staff.science.uva.nl/~zivkovic/FS2HSC/dataset.html`. The data set consists of about 2000 scans from which we select every 10^{th} scan. Thus, our test data set consists of 200 scans taken from a single robot in an apartment of size about $16 \times 8m$. As shown in fig. 7.10(a), the pre-alignment gained shows a huge error, additionally the alignment is very imprecise (blurred features). The experiment shows the power of FFS to escape local minima: starting with a large stepping parameter Δ_t, the first transformation blurs the data set and therefore weakens the wrong correspondences, giving space for new connections, fig. 7.10(b). Transforming all scans in parallel eventually results in a version of the map, which not only shows the misaligned parallel walls correctly contracted but also corrected the huge error as shown in fig. 7.10(d). The values of the parameters are equal to the experiment in section 7.4.2. Fig. 7.10 also shows a limit of the algorithm: one single initially strongly misaligned scan does not find consistent correspondences and therefore can not be correctly re positioned by the algorithm. It stays in its incorrect position. We assume that no algorithm working only on low level perceptual features is able to handle such a strong error correctly; mid level cognitive correspondences are needed. However, mid level perceptual features can easily be integrated into the system using correspondence functions modeling these perceptual forces, which will be part of the future work on the system.

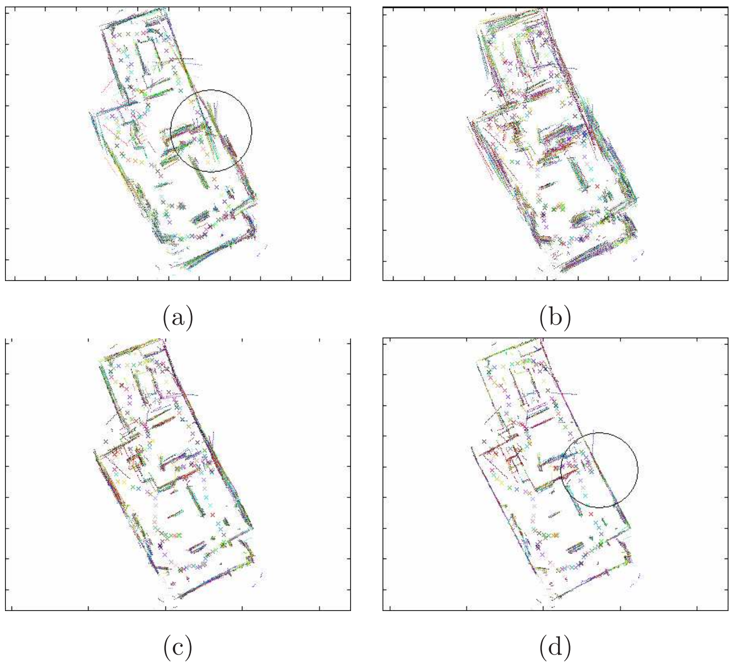


Figure 7.10: Apartment data set. (a) Initial configuration. The circled area shows an error due to incorrect loop closing. (b) A large step parameter dT blurs the map in the first iteration step to escape from the local minimum (c) Iteration 50: (d) Iteration 150: FFS has not only contracted the edges given in (a), but also has realigned the entire global map to fix the error (circled area).

7.4.4 NIST maze

This data set consists of 16 scans with similar structures, a typical indoor environment, yet again scanned with minimal overlap. See fig. 7.12 and fig. 7.13 for this experiment. The final result was achieved after 20 iterations, transforming the single scans up to 25cm and 20°. The size of the maze is $\approx 10 \times 10m$. FFS was able to align the scans correctly.

7.5 Conclusion and future work

We presented a new approach to the problem of multi robot mapping under the constraints given in rescue scenarios. It does not rely on odometry, i.e. the relative pose between the robots is unknown. It also can deal with the problem of extremely minimal overlap. Experiments conducted on a real data set of a disaster area from NIST shows the performance of the FFS approach under

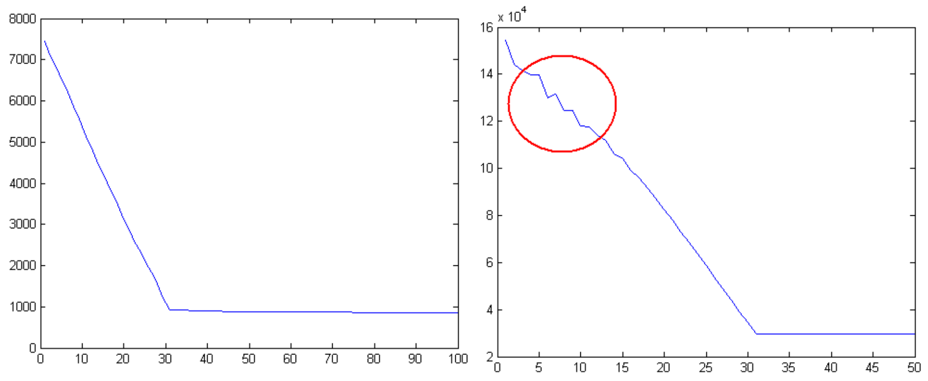


Figure 7.11: Left: potential vs. iterations of FFS for disaster data. Right: potential for Apartment data set. The potential (encircled) of the apartment data is not monotonically decreasing, indicating a possible escape from a local minimum

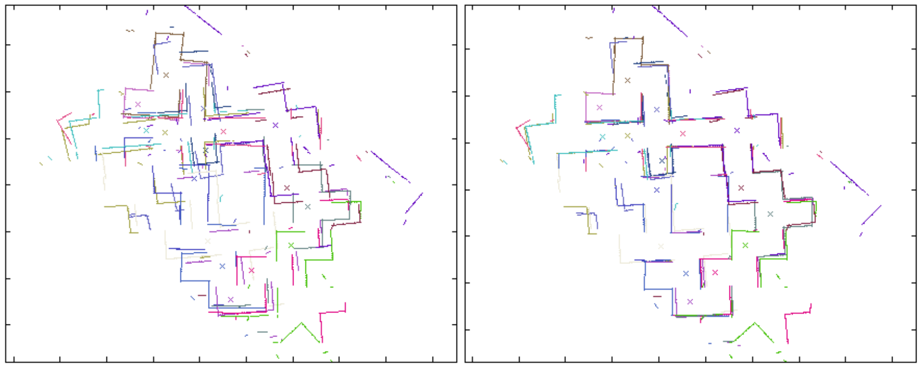


Figure 7.12: Left, initial configuration of NIST's maze data (16 scans). Right, after 5 iterations of FFS.

these complicated constraints and proved its applicability to the problem of multi robot mapping, they also proved the excellent performance of the algorithm correcting effects from wrong pre alignment. The future work will mainly focus on detection of higher level features: the modeling of the correspondence function with respect to the masses opens different ways to interface to mid level modules. The approach is easily extendable to $3D$, a report about the performance of an implementation of the $3D$ FFS is the topic of a forthcoming paper.

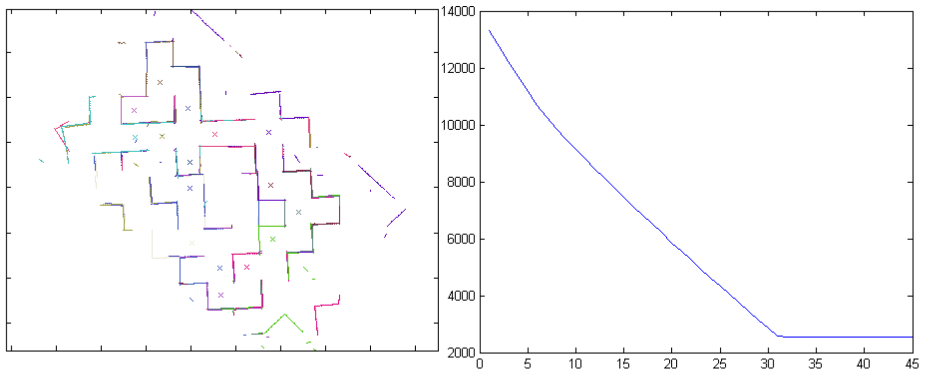


Figure 7.13: Left, final map obtained with FFS. Right, the potential vs. iterations plot.

Acknowledgment

We thank Andreas Nüchter from University of Osnabrück, Germany, for his helpful comments and the code for the experiments on $3D$-Lu/Milios-SLAM.

CHAPTER 8

Improving Sparse Laser Scan Alignment using Virtual Scans

We present a system to increase the performance of feature correspondence based alignment algorithms for laser scan data. Alignment approaches for robot mapping, like ICP or FFS, perform successfully only under the condition of sufficient overlap of features between individual scans. This condition is often not met, for example in sparsely scanned environments or disaster areas for search and rescue robot tasks. Assuming mid level world knowledge (in the presented case, weak presence of noisy, roughly linear or rectangular-like objects) our system augments the sensor data with hypotheses ('Virtual Scans') about ideal models of these objects. These hypotheses are generated by analyzing the current aligned map estimated by the underlying iterative alignment algorithm. The augmented data is used to improve the alignment process. Feedback between the data alignment and the data analysis confirms, modifies, or discards the Virtual Scans in each iteration. Experiments with a simulated scenario and real world data from a rescue robot scenario show the applicability and advantages of the approach.

8.1 Introduction

Robot mapping based on laser range scans has become a major field of research in robotics in the recent years. The basic task of mapping is to consistently combine spatial data, usually obtained from laser range devices, called 'scans', to a single data set, the 'global map'. The global map forms the basis for the task of robot localization. The map represents the environment scanned from different locations, even possibly scanned by different robots in the case of 'multi robot mapping'. The main problem in mapping is to correct for pose (position and heading direction) errors in scanning the environment. One class of solutions estimates rigid transformations (translations and rotations) of scans based on feature correspondences between the individual scans to correct for the pose errors. Techniques like ICP (Iterative Closest Point, e.g. [20, 205] and [183]) or FFS (Force Field Simulation based alignment, [166]) belong to this class. They show impressive results, but are naturally restricted: (1) Since they are based on feature correspondence, they require the presence of overlapping features among scans. (2) These algorithms depend on sufficiently good initialization to avoid local minima in the estimation of transformations. In this paper, we suggest a solution to the first problem: correct alignment in the absence of sufficient feature correspondences. This problem for example can arise in search and rescue environments (these environments typically show a small number of landmarks) or when a team of multiple robots builds a global map jointly. In this situation scans acquired from different views do not necessarily reveal the entire structure of the scanned object. The motivation to our approach is that even if the optimal relations between structures of scans is not known, it is possible to infer hypotheses of underlying structures from the sub-optimal alignment of single scans based on the assumption of certain real world knowledge. The hypotheses can then be used to improve the alignment. Fig. 8.1 illustrates the basic idea.

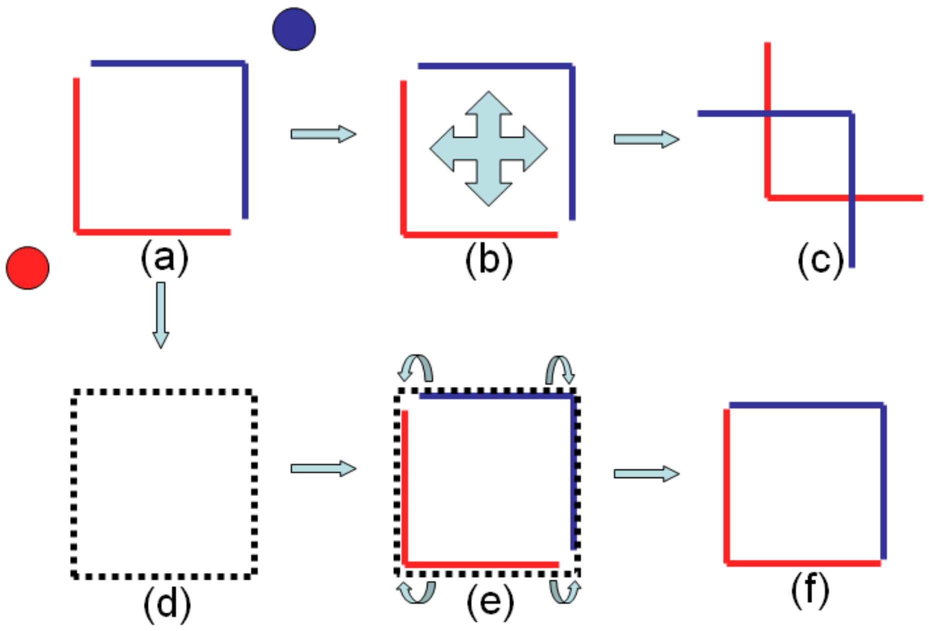


Figure 8.1: Motivation of the Virtual Scan approach. Top row shows alignment without Virtual Scans. Bottom row shows it with Virtual Scans. (a) Rectangular object scanned from two positions (red/blue robots). (b) Correspondences (block arrows) between the two scans (red/blue) does not reveal the scanned structure. (c) Misalignment due to wrong correspondences. (d) Analysis of estimated global map detects the rectangular structure (dotted rectangle). (e) The structure is added as a Virtual Scan. (f) Correct alignment achieved due to improved correspondences (curved arrows in (e)) between real world scans and Virtual Scans.

The top row shows a situation where the relations between features of scans can not reveal the real world structure, and therefore leads to misalignment. In the bottom row, analysis from a global view estimates the underlying structure. This hypothesis then augments the real world data set, to achieve a correct result.

The motivational example shows the ideal case; it doesn't assume any error in the *analysis* of global map in detecting the underlying structure. Our system also handles the non ideal situation where detecting structures and alignment becomes an iterative process. It utilizes a feedback structure between hypothesis generation and the scan alignment. The hypotheses are improved from the current alignment which are in turn used to improve the alignment. This will be discussed in more detail below. We first want to explain our approach in a more general framework.

Feature correspondence algorithms such as ICP or FFS can be seen as "Low Level Spatial Cognition" processes (LLSC), since they operate based on low level geometric information. The feature analysis of the global map, which is suggested in this paper, can be described as "Mid Level Spatial Cognition" process (MLSC), since we aim at analysis of mid-level features like lines, rectangles, etc. Augmenting real world data with expected models can be seen as an example of integration of LLSC and MLSC processes to improve the performance of spatial cognition tasks in robotics. We are using the area of robot perception for mobile rescue robots, specifically alignment of 2D laser scans, as a showcase to demonstrate the advantages of these processes.

In robot cognition, MLSC processes infer the mid level from low level data based on *global* properties of the data. In our case, we detect the presence of simple mid level objects viz. line segments and rectangles. The MLSC processes model world knowledge, or assumptions about the environment. In our setting for search and rescue environments, we assume the presence of (collapsed) walls and other man made structures. If possible wall-like elements or elements resembling rectangular structures are detected, our system generates the most likely model as a hypothesis, called a 'Virtual Scan'. Virtual Scans are represented using the same data format as the raw sensor data. Hence the low level alignment process can align the original scan data augmented by Virtual Scans.

In robot cognition, LLSC processes usually describe feature extraction based on *local* properties of the data (e.g. orientation of data point). In our system laser scans (virtual and real) are aligned into a global map using 'Force Field Simulation' (FFS) as the LLSC core process. FFS was recently introduced to robotics in [166]. In FFS, data points and correspondences are assigned weights, or values of certainty in an adaptive way. Hence FFS is a *natural* choice over its main competitor, ICP ([20, 205]), for using the Virtual Scans. The certainty values of the virtual data points can be used to indicate the strength of hypotheses.

FFS is an iterative alignment algorithm. The two levels (LLSC: data alignment by FFS, MLSC: data augmentation) are combined into a feedback structure as shown in Fig. 8.2. The following steps are repeated in each iteration:

- The FFS-low-level-instance pre-processes the data. Correspondences are based on low level features. The low level processing builds an estimate of the global map, which provides input for the mid level cognition module.

- The mid level cognition module analyzes the current global map, detects possible mid level objects present in the real world. These can be seen as suggestions which are fed back into the low level module as Virtual Scans. The low level system then aligns the augmented data for re-evaluation by the mid level system.

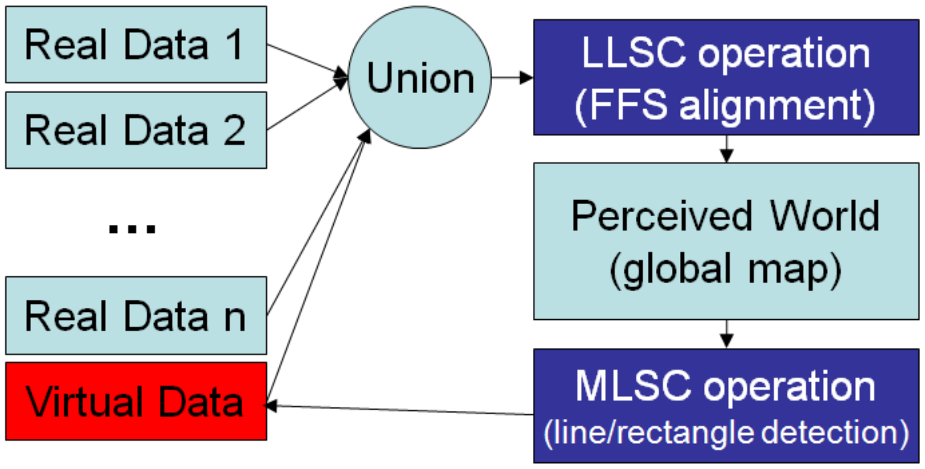

Real Data 1
Real Data 2
...
Real Data n
Virtual Data
Union
LLSC operation (FFS alignment)
Perceived World (global map)
MLSC operation (line/rectangle detection)

Figure 8.2: LLSC-MLSC feedback structure. The LLSC module works on the union of real scans and the Virtual Scan. The MLSC module in turn re-creates a new Virtual Scan based on the output of the LLSC module.

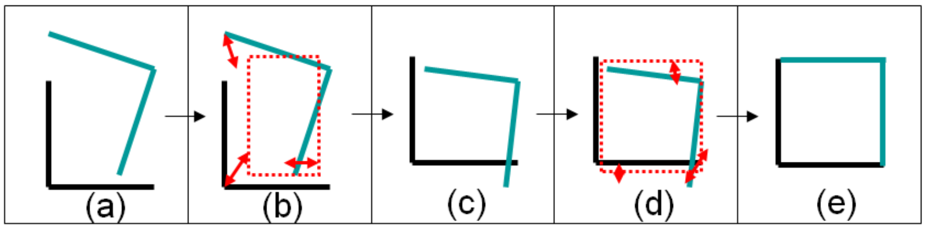


Figure 8.3: Feedback between Virtual Scans (VS) and FFS. (a) Initial state of real sensor data. (b) Real data augmented by VS (red rectangle). (c) After one iteration of FFS using real and virtual scans. (d) New hypothesis (red square) based on the map in (c). (e) Next iteration of FFS. Since this result resembles an ideal rectangle, adding an additional VS would not relocate the scans. Thus the system converged.

The following example will illustrate the feedback: Fig. 8.3 shows two scans from two different poses (see also Fig. 8.1). An MLSC process detects a rectangular structure (the assumed world knowledge) and adds a generating model (rectangle) to the data set. The LLSC module aligns the augmented data. The hypothesis now aligns the scans in an improved way. In each iteration, the aligned scans are analyzed to adjust the MLSC hypothesis. Thus LLSC and MLSC assist each other in a feedback loop.

8.2 Related Work in Spatial Cognition and Mapping

The potential of MLSC has been largely unexplored in robotics, since recent research mainly addressed LLSC systems. They show an astonishing performance: especially advances in statistical inferences [64, 118, 143] combined with geometric modeling of human perception [90, 73, 208] and the usage of laser range scanners contributed breakthroughs in robot applications, with the most spectacular results achieved in the DARPA Grand Challenges (2005 and 2007) where several autonomous vehicles were able to successfully complete the race [74]. Although sophisticated statistical and geometrical models like Extended Kalman Filters (EKF) [130], Particle Filters [118] and ICP (Iterative Closest Point) [20, 205] utilized in mapping approaches show impressive results, their limits are apparently clear in the aforementioned scenarios like search and rescue. These systems are still based on low level cognitive features, since they construct metric maps using correspondences between sensor data points. Having these well-engineered low level systems at hand, it is natural to combine them with MLSC processes to mutually assist each other and obtain improved performance in broader applications.

The study of MLSC in humans, in particular in spatial intelligence and learning, is advancing rapidly [96, 150, 257]. Such results are used to model generic representations of space for mobile robots using both symbolic ([156]) and non symbolic ([104]) approaches. Naturally, SLAM (Simultaneous Localization and Mapping) which is a spatial cognitive process [58] is often used as an example application [189]. In [261], a spatial cognition based map is generated based on High Level Objects. Representation of space is mostly based on the notion of a hierarchical representation of space. Kuipers [156] suggests a general framework for a Spatial Semantic Hierarchy (SSH), which organizes spatial knowledge representations into different levels according to ontology from sensory to metrical information. SSH is an attempt to understand and conceptualize the cognitive map [158], the way we believe humans understand space. More recently, Yeap and Jefferies [280] trace the theories of early cognitive mapping. They classify representations as being space-based and object-based. Comparing to our framework, these classifications could be described being related to LLSC and High Level Spatial Cognition (HLSC), hence the proposed LLSC-MLSC system would relate closer to space-based systems.

In [19], the importance of 'Mental Imagery' in spatial cognition is emphasized and basic requirements of modeling are stated. Mental Images invent or recreate experiences to resemble actually observed events or objects. This is closely related to the "Virtual Scans" described in this proposal. Recently, Chang et al. [47] presented a predictive mapping approach (P-SLAM), which analyzes the environment for repetitive structures on the LLSC level (lines and corners) to generate a "virtual map". This map is either used as a hypothesis in unexplored regions to speed up the mapping process or as an initialization help for the particle filter when a region is first explored. In the second case the approach has principles similar to those in using Virtual Scans. The impressive results of P-SLAM can also be seen as proof of concept of integrating prediction into robot perception.

The problem of geometric robot mapping is based on aligning a set of scans. On the LLSC level the problem of simultaneous aligning of scans has been treated as estimating sets of poses [183]. There are numerous image registration techniques, the most famous being Iterative Closest Point (ICP) [20], and its numerous variants to improve speed and convergence basins. All these techniques essentially perform a search in transformation space (translations and rotations) to find the set of optimal pair-wise transformations of scans. They optimize some objective function defined on transformation spaces. The techniques vary in defining the optimization functions that range from being error metrics like "sum of least square distances" to quality metrics like "image distance" [23]. 'Force Field Simulation' (FFS) ([166]) minimizes a potential derived from forces between corresponding data points in all the scans. The Virtual Scan technique presented in this paper will be integrated with FFS as underlying alignment technique.

8.3 Scan Alignment using FFS

Understanding some aspects of FFS is important for understanding the proposed extension using Virtual Scans. We will give an overview of FFS here. FFS aligns scans obtained by robots, typically from different poses. We assume the scans to be roughly pre-aligned (see Fig. 8.11(b)), using odometry or shape based pre-alignment [4]. This is in accord with the performance comparison between FFS and ICP described in [164]. In FFS, each scan is seen as a non-deformable entity, a 'rigid body'. In each iteration, a translation and rotation is computed for all scans *simultaneously* [165]. This process minimizes an objective function defined on the set of all data points (real and Virtual Scans). The optimization is performed using a gradient descent approach motivated by simulation of dynamics of rigid bodies (the scans) in gravitational fields, but "*replaces laws of physics with constraints derived from human perception*" [166]. The gravitational field is based on strength of correspondences (forces) between all pairs of data points. The forces are designed in such a way that a low overlaying potential corresponds to a visually good appearance of the global map. FFS minimizes the potential induced by the forces and converges towards a local minimum of the potential, representing a locally optimal configuration of scans. The scans are translated and rotated according to the laws of motion of rigid bodies in a force field. Fig. 8.4 shows the basic principle: forces (red arrows) are computed among four scans. FFS simultaneously transforms all scans until a stable configuration is reached.

If S_1, S_2 being two different scans, the force between two data points $v_i \in S_1$ and $u_j \in S_2$ is defined as a vector

$$V(v_i, u_j) = M(v_i, u_j)\frac{v_i - u_j}{\|v_i - u_j\|} \tag{8.1}$$

Its magnitude is defined as:

$$M(v_i, u_j) = \frac{1}{\sigma_t\sqrt{2\pi}} e^{\left(-\frac{\|v_i-u_j\|^2}{2\sigma_t^2}\right)} w_i w_j \cos(\angle(v_i, u_j)) \tag{8.2}$$

where $\sigma_t,\ w_i,\ w_j,\ \angle(v_i, u_j)$ defined as follows: $\angle(v_i, u_j)$ denotes the angle between the orientations of points, which is defined as the angle between orientations of the underlying locally linear structures. See Fig. 8.5(a) for an example, which especially shows the influence of the cosine term in Eq. (8.2): forces are stronger between parallel structures. σ_t is a parameter steering the radius of influence of force exerted by a data point on other data points. σ_t is decreased during the iterative process as the alignment improves, thus changing the influence of each data point to be more local. The weights w_i, w_j express the certainty of the points v_i, u_j. They can model the "feature importance" of a data point. We utilize this feature of FFS to model the strength of hypothesis in the Virtual Scans. Hence in Eq. (8.2) the interaction between LLSC and MLSC can be seen directly: the distance ($\|v_i - u_j\|$) and cosine terms correspond to LLSC, while the weights are derived from MLSC (in our case the Virtual Scans). Computing the total force in a näive way has a time complexity of $O(m^2)$ where m is the number of data points in a particular configuration. But using 'neighborhood approximations' and 'Fast Gauss Transforms' the complexity can be reduced to $O(m \log m)$ or $O(m)$. Please see Section IV of [165] for details.

The motion updates of the scans are computed using Newton's laws of motion by applying the forces for Δ_t units of time. Then the forces are re-computed by analyzing the new configuration of the scans. For a single transformation step see Fig. 8.5(b). Δ_t can be seen as the step width in the gradient descent process. It is monotonically decreased, allowing the system in early iterations to jump out of local minima, yet to be attracted by local features in later steps for convergence. The interplay between σ_t and Δ_t is an important feature of FFS. See Figs. 8.11 and 8.12 for an example of the performance of FFS on a laser range data set.

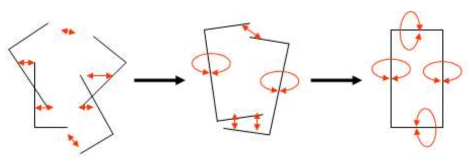


Figure 8.4: Basic principle of FFS. Forces are computed between four scans. Red arrows illustrate the principle of forces. The scans are iteratively (here: two iterations) transformed by translation and rotation until a stable configuration is achieved.

FFS is closely related to *simultaneous* ICP. A performance evaluation of both algorithms showed similar results [164]. In general, FFS can be seen as more robust with respect to global convergence with poor initialization, since the point correspondences are built not in a hard (nearest neighbor) way but in a soft(sum of forces) way. Also the inclusion of weight parameters makes it a

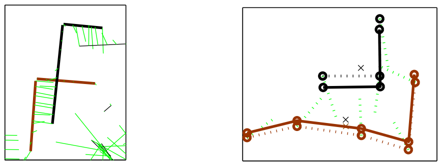


Figure 8.5: FFS example: Showing forces and update. (a) Forces (green) between two rigid structures in two different scans (brown, black). The black and brown lines connect the actual data set for display reasons only. The figure shows a magnification of the upper left corner of Fig. 8.11(b). (b) Example of force and movement. Dotted lines show two scans (black, brown) and their forces (green) in iteration t. Solid lines show the resulting transformed scans at iteration $t + 1$.

natural decision for our purpose of extension using Virtual Scans.

8.4 Creating Virtual Scans: Mid Level Analysis

The laser range data goes through two pre-processing steps before the scans are aligned. First, the underlying local linear structures (line segments) are detected in each scan. Classic approaches like Hough line detection rely on local linearity of the underlying data points. Hence such techniques are very sensitive to noise. We therefore use a recently published technique [170] which is specifically tailored to model laser scan data with line segments, using extended Expectation Maximization. Second, data points are equidistantly sampled along these line segments. The original data is discarded in favor of the newly sampled points. This solves certain problems of noisy scans. It also reduces the number of points drastically. See Fig. 8.6 for an example. In each iteration

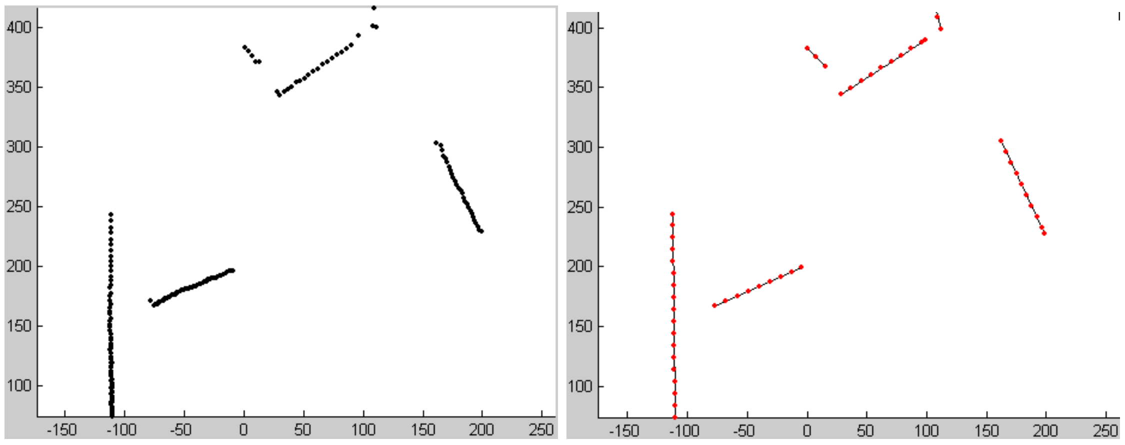


Figure 8.6: Pre-processing steps in FFS. Left: Original scan. Right: Line segments (black) and re-sampled data points (red).

the mid-level analysis is performed on the current estimate of the global map (without the Virtual Scans). It involves detecting linear and rectangular structures as explained below.

8.4.1 Lines

The usage of linear structures for our Virtual Scan approach is motivated by the world knowledge assumption of scanning a man made environment (e.g. a collapsed house): although these environments often locally don't show major linear elements any longer, a global view often reveals an underlying global linear scheme, which we try to capture using a global line detection. Here (in contrast to our *local* line segment detection in the pre-processing step) we use the classic line detection approach of Hough transform [127], since it detects *globally* present linear structures. Hough transform does not only show location and direction of a line, but also the number of participating data points. We use this value to compute a certainty-of-presence measure, i.e. the strength of the line hypothesis. The data points on these structures inherit this certainty as their weights (e.g. w_i, w_j in Eq. (8.2)). We only use lines above a certain threshold of certainty. We will specify how the detected lines are utilized to create the Virtual Scan in Section 8.4.3. The Hough detection is performed on the entire set of (re-sampled) data points. We do not use the local linear information of the line segment representation of the data here, since we aim at global linearity. This is more robustly detected by Hough transform.

8.4.2 Rectangles

The rectangles are detected using the entire set of local line segments of all scans, obtained after the first pre-processing step. We use the rectangle detection approach described in [160]: each line segment (of each scan) is translated into 'S, L, D space' (Slope, Length, Distance), which simplifies the detection of appropriate (rectangular like) configurations of four near parallel and near perpendicular segments. For details see [160]. Before detecting rectangles we merge spatially close lines in a cluster to a single prototype line using a line merge approach described in [167] as shown in Fig. 8.7(b). The rectangle detection module then predicts location, dimension and certainty-of-presence of hypothetical, rectangles present in the data set of merged line segments. The certainty, or strength of the hypothesis is derived from properties (segment length, perpendicularity) of line segments participating in generating rectangles. This value is used to create the weight of the rectangle (and data points on it) in the Virtual Scan.

8.4.3 Creating a Virtual Scan

A Virtual Scan is a set of virtual laser scan points. The detected line segments and rectangles are represented as point sets in a Virtual Scan as if detected by a virtual laser scanner. All detected MLSC elements (lines, rectangles) are represented in a single Virtual Scan. A sample

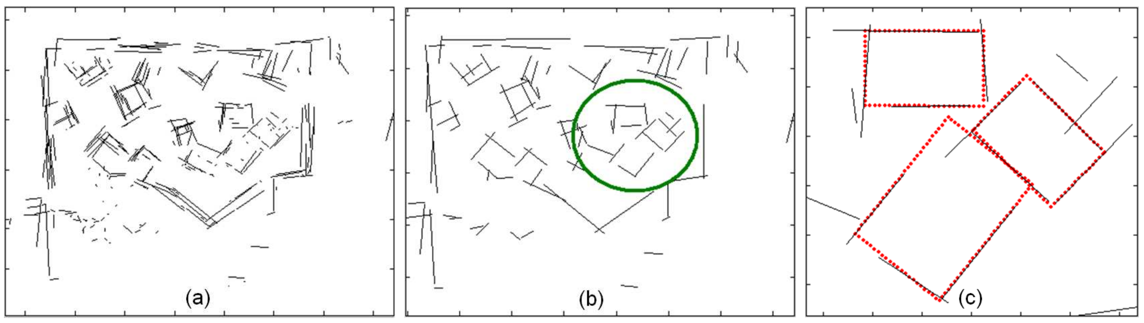


Figure 8.7: Rectangle detection. (a) Global map built by line segments of all individual scans. (b) Result of global line merging. (c) Magnification of area encircled in (b). The detected rectangles are shown in red.

Virtual Scan is shown in Fig. 8.8(b).

An important feature of the Virtual Scan is that each of its point is assigned a weight, representing the strength of the hypothesis of the underlying virtual structure. Utilizing this feature, we benefit from the weights that steer the FFS alignment. As defined in Eq. (8.2), the weights w_i, w_j directly influence the alignment process: stronger points, i.e. points with higher value of w, have a stronger attraction. Hence, a strong hypothesis translates into a locally strongly attractive structure. The strength of a hypothesis reflects the belief of the hypothesis relative to the real data. All data points of the real scan are assigned a 'normal/uniform' weight of 1.

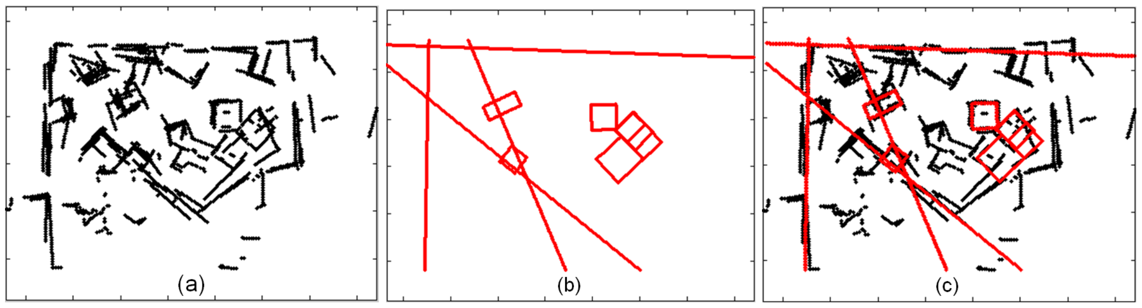


Figure 8.8: Virtual Scans in an early stage of FFS. (a) Global map. (b) The Virtual Scan consisting of points representing detected linear and rectangular structures. (c) superimposition of real scans and the Virtual Scan. This is the data used in the following iteration of FFS.

8.5 Alignment using Virtual Scans: Algorithm

The algorithm 2 describes the interaction between LLSC (FFS) and the MLSC processes. S_i, $i = 1..n$, denotes the real scan data, consisting of n scans. $V^{[t]}$ denotes the Virtual Scan in iteration t.

Algorithm 2 Force field based mapping using Virtual Scans

```
1: Create set of line segments $L_i$ for each scan $S_i$ using the technique in [170].
2: Init: $t = 1$, $V^{[0]} = \emptyset$.
3: repeat
4:     Perform FFS on $\bigcup_{i=1..n} S_i \cup V^{[t-1]}$, resulting in transformations (translation, rotation) $T_i^{[t]}$ for each scan $S_{i=1..n}$ (Section 8.3).
5:     Construct a global map $G$ and a set of global $GL$ of line segments by transforming the scans and their line segments into a common global frame: $G = \bigcup_{i=1..n} T_i^{[t]}(S_i)$; $GL = \bigcup_{i=1..n} T_i^{[t]}(L_i)$.
6:     Detect set of lines $\mathcal{L}$ in $G$, set of rectangles $\mathcal{R}$ in $GL$ (Sections 8.4.1,8.4.2).
7:     Create Virtual Scan $V^{[t]}$ using the scan points representing the elements of $\mathcal{L}$ and $\mathcal{R}$ (Section 8.4.3).
8:     Update $\sigma_t$ and $\Delta_t$ for the FFS process (Section 8.3).
9: until FFS converges to a stable global map
```

8.6 Experiments and Results

8.6.1 Sparse Scanning (Simulated Data)

This experiment shows the effect of Virtual Scans in a sparsely scanned environment. It features a simple environment to highlight the principle of Virtual Scans and to show the improvement in the alignment process. Please compare this to the motivational example introduced in Section 8.1, as well as to Figs. 8.1 and 8.3. Fig. 8.9 shows a simulated arena consisting of four rectangular rooms scanned from five different positions (shown as 'x' marks). Each scan is pre-processed and randomly translated and rotated to simulate pose errors. We first try to align this data set using FFS without Virtual Scans. The performance of FFS depends on the initial value of σ_t (Eq. (8.2)) i.e. σ_0. We tried multiple initial values: results of $\sigma_0 = 30$ and $\sigma_0 = 80$ are shown in the bottom row of Fig. 8.9 left and right respectively. σ is measured in units of the data set. The width of the simulated arena is 400 units. In bottom left, with a lower σ_0, local structures are captured and aligned correctly, but global correspondences can not be detected (the 'hallway' between the rooms is offset incorrectly). As can be seen in bottom right, increasing σ_0 and thereby strengthening the influence of global structures leads to wrong results since local correspondences become relatively unimportant: FFS optimizes correspondences of major structures (although they are far from each other in the initial map). The disability of balancing the influence of local and global structures is an inherent drawback of alignment processes which are based on point correspondences (e.g. ICP, FFS), and not a special flaw of FFS only (other values of σ did not improve the alignment either).

Fig. 8.10 shows the improvement when Virtual Scans are used. Here we use $\sigma_0 = 30$). FFS is able to detect the correct local structures, and the global structures are captured through augmentation by Virtual Scans. Also the effect of the hypothesis adjustment by feedback is clearly

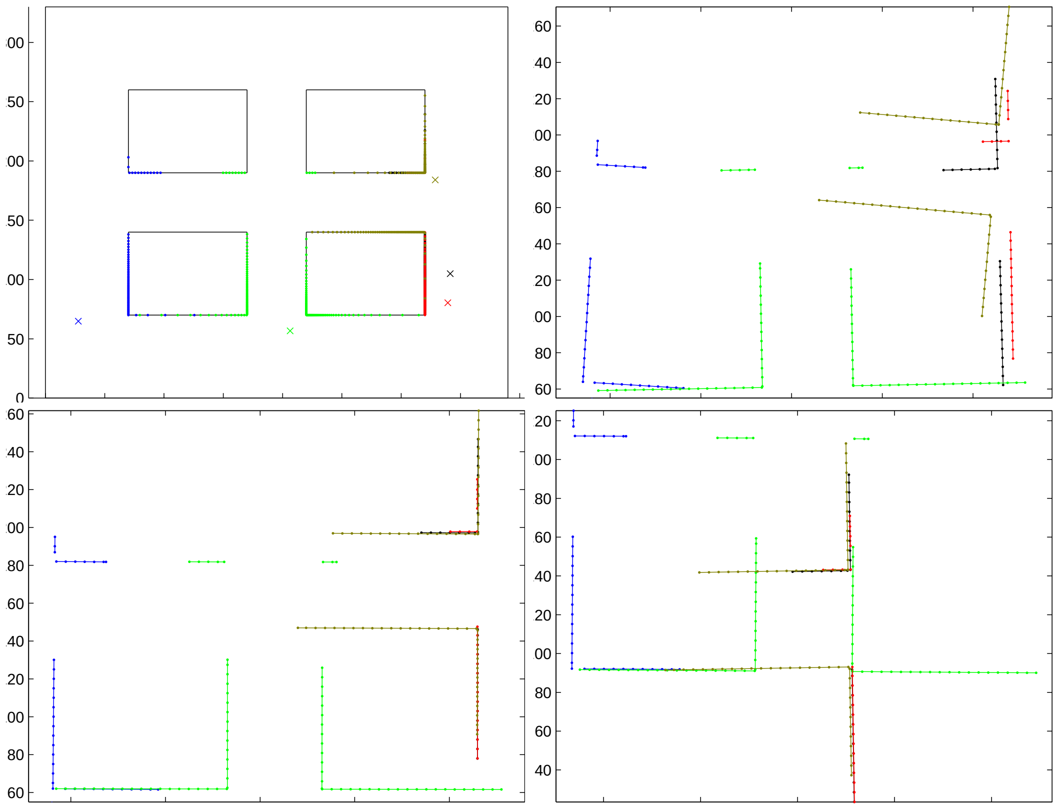


Figure 8.9: Top row. Left: Simulated arena consisting of four rooms scanned from five different positions (crosses). Points of same color belong to a single scan. Right: Pre-processed scans after adding random pose errors. Bottom row. Results of FFS without Virtual Scans. Initialized with configuration in top right. Left: $\sigma_0 = 30$. Right: $\sigma_0 = 80$.

visible in Fig. 8.10: Top left shows an early hypothesis, which contains a wrong rectangle and misplaced lines. This early hypothesis is corrected by the feedback process between FFS and the rectangle/line detector. Top right shows a later iteration, the line position is adjusted (though not perfect yet), two rectangle hypotheses compete (lower right corner). The final result is shown in the bottom row. The detected lines adjusted expected global structures (the walls of the 'hallway') correctly, the winning rectangle hypothesis 'glued together' the corners of the bottom right room. Please notice that this room is a structure that is not entirely present in any individual scan, but only detectable in the global map. Hence only the Virtual Scan enhanced FFS could perform correctly.

8.6.2 NIST Disaster Area

This experiment shows the improved performance of the alignment process on a real world data set. The data, provided by National Institute of Standards and Technology (NIST) represents a simulated indoor disaster scenario. The data set consists of 60 single laser scans, taken from 15

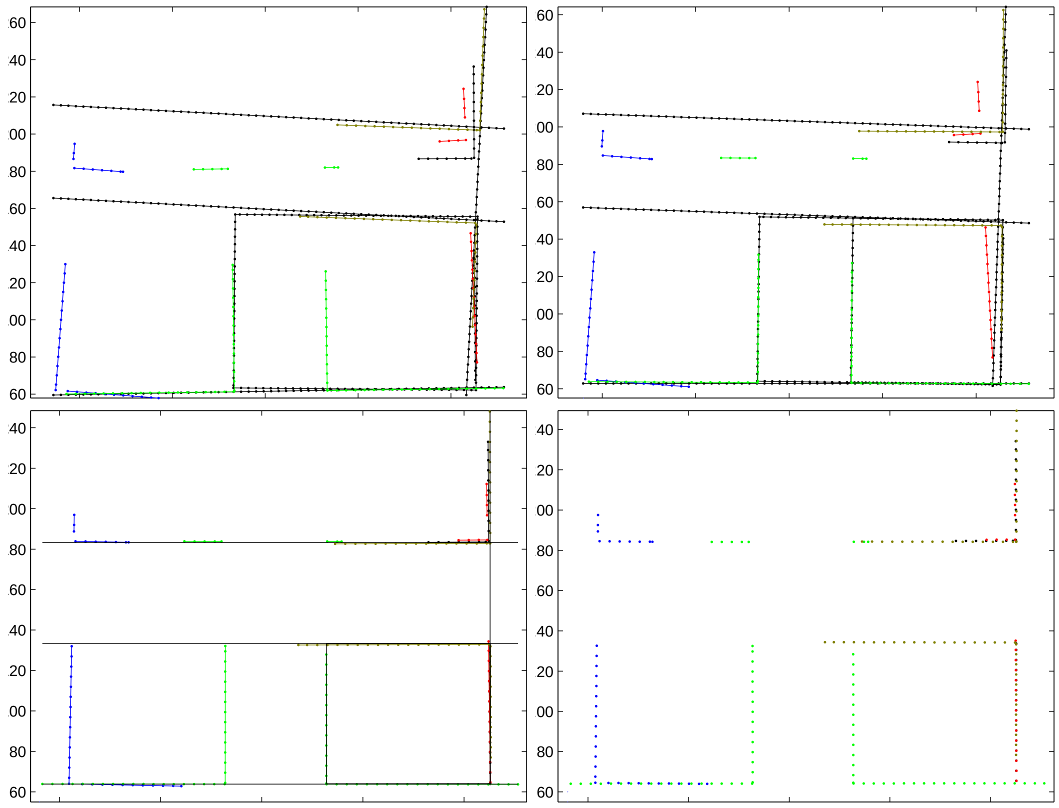


Figure 8.10: Same experiment as in Fig. 8.9, but using FFS with Virtual Scans and $\sigma_0 = 30$. Elements of Virtual Scans (lines and rectangles) are shown in black. Top row. Left: After iteration 10. Right: After iteration 20. Bottom row. Left: After iteration 30 (final result). Right: Same as that in left, but the Virtual Scan is not shown.

different positions in 4 directions (N,W,S,E) with 20° overlap. It can be interpreted as a scene scanned by 15 robots, 4 scans each. No particular order of scans is assumed. The scans have little overlap and have no distinct landmarks. The initial global map was computed using a shape based approach described in [4]. See Fig. 8.11 for sample individual scans and the initial global map. We used the initial global map for two different runs of FFS, one using Virtual Scans, one without Virtual Scans. The increase in performance is qualitatively evaluated by visual inspection, since the ground truth data is not available. As can be seen in Fig. 8.12 the utilization of Virtual Scans leads to distinct improvement in overall appearance and mapping details. Overall, the map is more 'straight' (see for e.g. the top wall), since the detection of globally present linear structures (top and left wall in Fig. 8.12) adjusts all participating individual segments to be collinear. These corrections advance into the entire map. The improvements can be better seen in certain details: the most distinct improvements are encircled in Figs. 8.12(d) and (f). The rectangle in the center of the global map is an excellent example for a situation where correct alignment is not achievable with only low level knowledge. Only the rectangular structures from the Virtual Scan (see Fig. 8.12(c)),

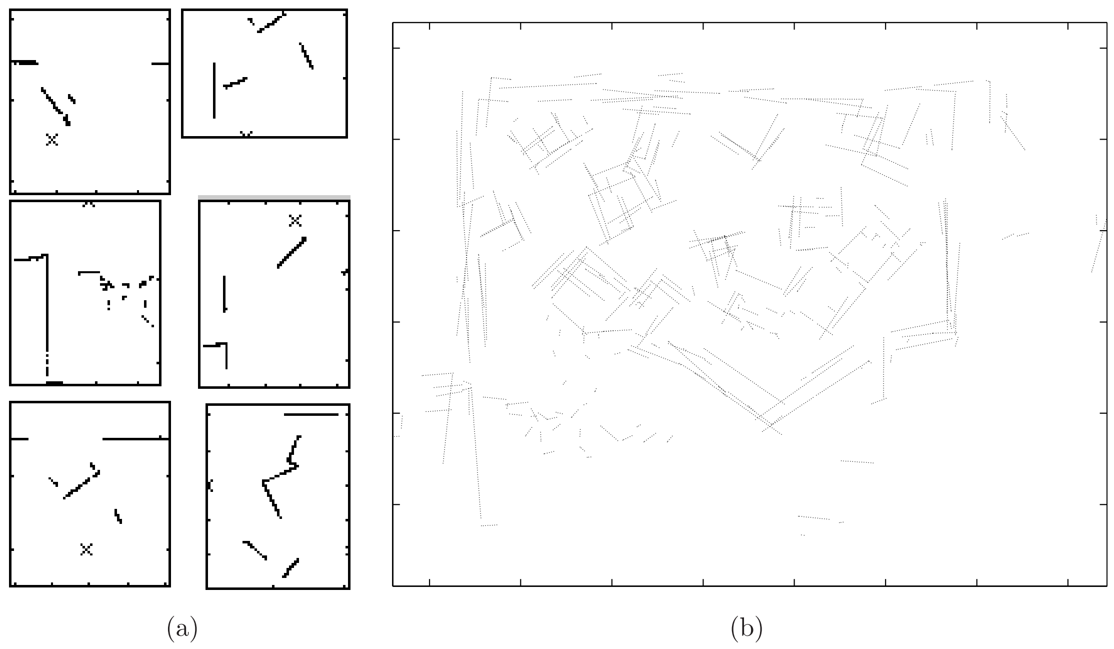


Figure 8.11: The NIST disaster area data set. (a) 6 example scans (from a total of 60). Crosses show each robot's position. (b) 60 scans superimposed using a rough pre-estimation using shape similarity among scans [4]. This is the initial global map for the experimental results shown in Fig. 8.12.

can force the low level process to transform the scan correctly. Without the assumed rectangle the low level optimization process necessarily tries to superimpose the two parallel sides of the rectangle to falsely appear as one. Magnification of both situations can be seen in 8.12(e).

8.7 Conclusion and Outlook

The presented implementation of an extension to the FFS alignment process using Virtual Scans could significantly improve the results for the alignment task. The Virtual Scans are composed of hypothetical mid level real world structures. The implementation provides a proof of concept of the applicability of the presented concept for the combination of LLSC and MLSC processes. The detection of simple elements (lines, rectangles) based on weak real world assumptions could improve the performance. We are aware that adding domain knowledge certainly enhances the risk of wrong inferences. The proposed system handles errors caused by premature belief in mid level features by implementing the feedback principle, which evaluates a single hypothesis. It is known that single hypothesis systems introducing higher knowledge tend not to be robust. Under certain circumstances this behavior could also be observed in experiments with our system, which needed manual parameter adjustment to steer the influence of the hypotheses. The combination can be embedded into a multiple hypotheses framework, e.g. particle filters, which will be part of future work.

Additional future work involves determining the level of elements at which we can have meaningful enough elements to improve the alignment process, yet not be over biased. The current elements were chosen to model assumptions of indoor disaster areas. For outdoor city scenarios, these elements along with additional geometric structures (triangles and ellipses for trees etc.) might also be helpful. For more general scenarios, our research will not assume fixed elements, but will incorporate learning of distinct, repetitive basic elements to build a 'shape alphabet'. These elements will be defined using general poly lines, focusing mainly on shape description.

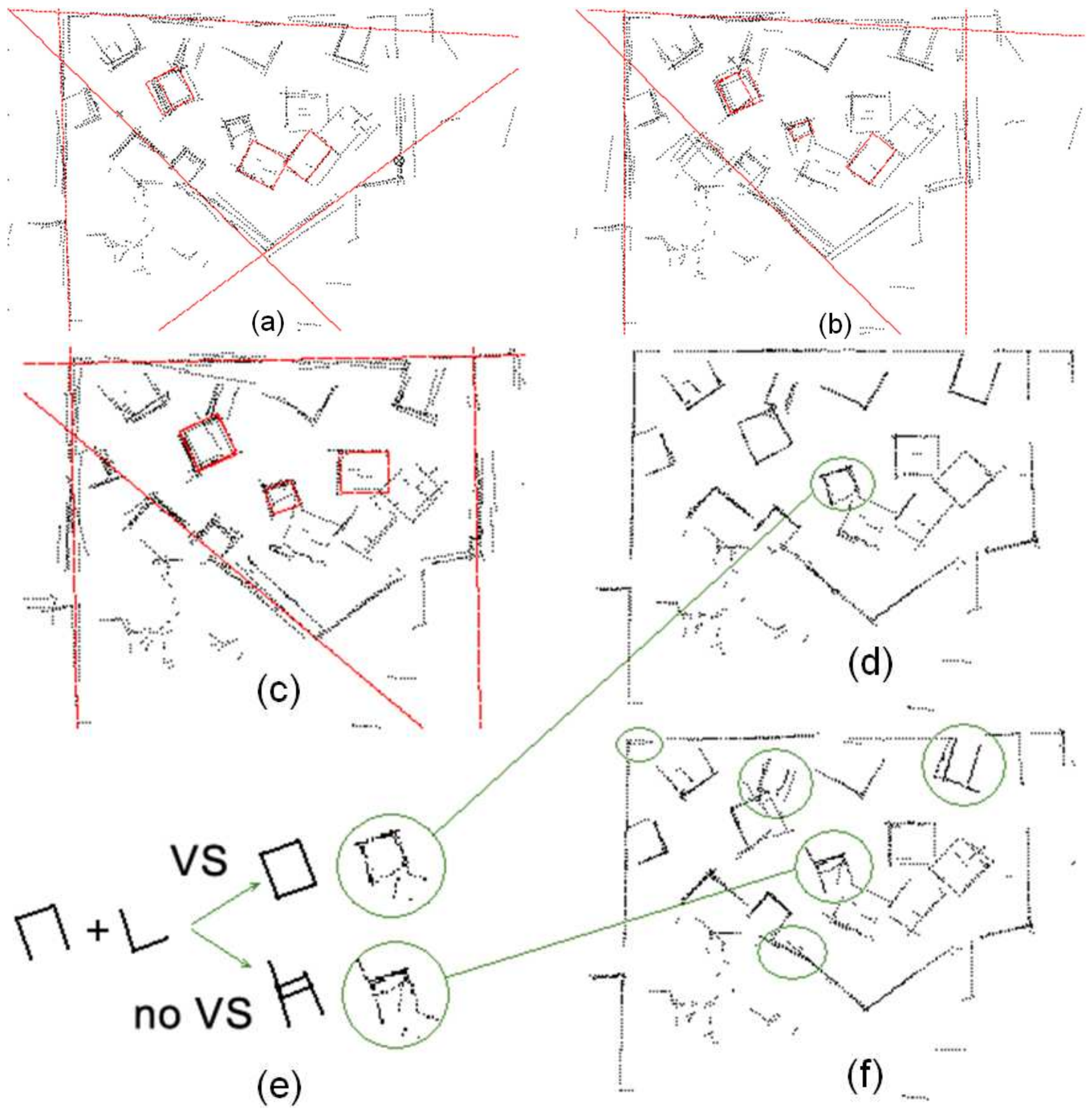


Figure 8.12: Alignment of NIST data set (initial alignment is shown in Fig. 8.11(b)) after (a) Iteration 4. (b) Iteration 8. (c) Iteration 10. In (a-c) red lines show the data of the Virtual Scan (VS). Please notice the mutual adjustment of hypotheses and real data. (d) Final result using Virtual Scans, after 100 iterations. The VS is not shown for clarity of display. (f) Final result of alignment without VS. Encircled areas show examples of improvement of (d) over (f). (e) The rectangle in the center could only be aligned correctly using the Virtual Scan. Please compare (e) to motivational examples in Figs. 8.1 and 8.3.

CHAPTER 9

Merging Maps of Multiple Robots using ViRtual-SLAM

Merging local maps, acquired by multiple robots, into a global map, (also known as map merging) is one of the important issues faced by virtually all cooperative exploration techniques. We present a novel and simple solution to the problem of map merging by reducing it to the problem of SLAM of a *single* "virtual" robot. The individual local maps and their shape information constitute the sensor information for the virtual robot. This approach allows us to adapt the framework of Rao-Blackwellized particle filtering used in SLAM of a single robot for the problem of map merging.

9.1 Multi-robotics

As the techniques for solving problems associated with single autonomous robots mature, the natural step ahead is to focus on autonomous multi-robot systems (MRS). There are several *potential* advantages of using a team of robots compared to using a single robot like fault tolerance, speed of exploration, better reactivity and deliberation. Of course there are a lot of issues that need to be addressed to actually *realize* the potential. It is also important to note that MRS are not always superior choice to single robot systems. There are some tasks that *require* team work (e.g. team of robots playing soccer), some that *benefit* but do not require (e.g. team of robots mapping an environment can speed up the process), while for some other tasks MRS are not desirable at all. Typically such decisions involve estimating overhead to benefit ratio. If it is high then MRS are not desirable. [132, 72, 66] present a nice overview of such discussions and also classify the MRS based on their various properties.

9.1.1 Exploration & mapping

One of the key challenges for autonomity of MRS is to cooperatively explore and build the world model of the environment they are in. A major successful and popular approach for such tasks is to actively or passively reduce uncertainties by using posterior representation of poses (localization) and world model (mapping). [251] provides an excellent exposition of such probabilistic techniques. Broadly speaking there are three main schemes of reducing uncertainties for MRS as explained below

1. **Reducing each others' uncertainties collectively in a shared world model:** The techniques used for single robots can be easily extended for MRS in this scheme since the robots *start* in a shared world model that is know their relative oriented positions (poses) and the data collected by several robots can be treated as if they were collected by a single robot [248, 85, 222, 218]. Also they can *actively* explore by using market architecture as described in [292] or by trying to maximize "utility" by minimizing the potential for overlap in "information gain" amongst various robots [236, 41, 40].

2. **Reducing each others' uncertainties collectively maintaining independent world models:** Under this scheme the robots do not need to start in a shared world model. They main joint posteriors of their poses and world models but update them independently and when any two robots detect each other they use mutual data to further improve their uncertainties of poses and/or world models.

   [93] uses Monte Carlo Localization (MCL) by approximating the joint posteriors of poses of multiple robots using factored representation of posteriors of poses of individual robots. Since they maintain posteriors of only poses their technique is called multi-robot *localization*. [144] represents posteriors of individual robot poses both *inside* and *outside* of their world models using particle filters and when two robots detect each other they update the posteriors so that they can get relative poses of the robots in each others' maps and merge their maps into a shared world model. They do this merging pair-wise and hence if there are k robots one needs k^2 particle filters. [243] extends this technique by improving on how the posteriors of poses outside their world models are updated. Instead of using a fixed likelihood as in [144] they use likelihood proportional to Dirichlet prior learnt from "typical" structured world models. [94] summarizes the above two papers in a nice expository style.

   [275] extends the Constrained Local Submap Filters (CLSF) of [274] for MRS. [274] improves the computational complexity of Kalman Filter based SLAM by maintaining estimates of world-model in local frame of the robot and the estimates of the position of the local frame in a global frame and fusing the local estimates into global estimates at appropriate intervals.

Since the local estimates can be decorrelated from global estimates as shown in [273], they can reduce the update complexity from $O(n^2)$ to $O(n_L^2)$ where n is the total number of features in a map (e.g. # of cells in case of an occupancy grid representation) and n_L is the number of locally detectable features by the robot. Typically $n_L << n$. The key challenge in using CLSF for MRS is to estimate the poses of the local frames in the global frame when relative poses are not known. [275] uses the Maximal Common Subgraph (MCS) technique of [15] to get the correspondences between the local world-models and use those correspondences to get the intial estimates for the relative poses of the local world models.

[248] uses Sparse Extended Information Filters (SEIF) to represent the joint posteriors of both poses and world models. SEIF gives them computational advantage over Extended Kalman Filters (EKF) and has nice properties that allows them to update the posteriors of the multiple robots independently without approximation as in [93]. When two robots detect each other in addition to updating posterior of poses using each others' data, they also merge the individual local maps to build a shared model. The merging is done is a very basic way similar to classical ICP [20]. [129] uses Rao-Blackwellized particle filters to represent the joint posteriors, which is superior to SEIF in that it allows for non-Gaussian noise in the data but is computationally much more expensive because of the high-dimensionality of joint posteriors.

3. **Reducing uncertainties independently (in a decentralized way) and then building a shared model:** In this scheme robots neither need to start in a shared world model nor maintain joint posteriors. Instead each robot maintains posteriors of its position and local world model in it's local frame and at appropriate intervals the local world-models are merged into shared (global) world model. The individual robots can build their local maps using any standard technique. The key problem (also called as map merging problem [146]) that needs to be addressed here is how to merge the local maps into a global frame. It is worthwhile to note that the second scheme also addresses this problem but is not the main focus of the scheme. This and the second scheme are more interesting in multi-robotics case because of more realistic assumptions. The scalability of MRS that is the number of individual robots used is more serious issue under the second scheme compared to the third. Our solution falls under the this scheme and we present results that could scale upto 16 robots. In contrast the techniques like [129, 93] which fall under second scheme could show results upto 4 and 2 robots respectively. Since the map merging problem is the key challenge that we address, problem definition assumptions and existing work on it are presented in section 9.1.2.

9.1.2 Map merging

- **Problem definition & assumptions:** The key challenge in merging different local maps into a global map is to compute the relative coordinate frames. The problem is difficult (and interesting) because no initial estimates of the relative poses of the frames are assumed. Of course one underlying very weak assumption is that the solution exists and can be *verified/evaluated*. Another common assumption about the verification process itself is that it can be done using the "good and consistent alignment perceptually" heuristic.

- **Existing work:** In virtually all existing work the problem is treated as a *search* problem by iteratively proposing candidate configurations and verifying the quality of the proposal. They all differ in what factors guide their proposal and verification processes.

  [192] uses omnidirectional vision and merge maps when two robots "run into" each other which gives them the proposal. They verify it using their color based matching of the local features of the omnidirectional images of the robots. [146] present a decision-theoretic way of verifying the proposal. They use scan-matching of "patches" (features like corridors, corners etc.) of the local maps to compute likelihood of the proposal. They refer to [202] for the proposal process. Their scan-matching is based on correlation operator ([148]) and use "map smearing" to take into account the uncertainties in scan readings. [57, 163, 4, 7] are all based on segment based maps and use different "segment based similarities" for proposal and their "alignment-heuristics" to verify them. These work well in their presented cases but do not provide uncertainties of their processes. [131] uses a higher abstraction representation like topological maps that gives them the power to use graph matching algorithms ([15, 38, 276]) for proposals. They use clustering combined with heuristics to evaluate the proposal. They also use a data structure based multi-hypotheses tracking to be able to recover from bad local greedy merges.

  [43] uses more advanced tools motivated from motion planning literature like the seminal work on "probabilistic road maps" [141] and its numerous varied applications [238, 6, 172]. They designed an adaptive random walk based motion planning ([45, 44]) and use it for proposal and use special image dissimilarity metric based on [24] for not only verifying but also adapting the sampling distribution involved in their random walk based proposal. Though their approach is superior to most of the existing approaches their main drawback is that their random walks assume gaussian distributions for convergence and if the initial proposal places local maps too wide apart in the global frame, their metric won't be able to modify the distribution appropriately and hence might not converge. Our approach is in fact based on extension of their metric and replaces their motion planning based approach with SLAM of a virtual robot that allows us to exploit the power of existing SLAM algorithms.

In the proposed approach we employ Sequential Monte Carlo estimation of transformations in the merging process by tracking multiple hypotheses. The estimation process is guided by shape information in the local maps. The main difference between straightforward multi-robot SLAM as in [129, 93] and our ViRtual SLAM (VR-SLAM) is that we restrict the state space to that of a trajectory of the single (virtual) robot. As a consequence we do not estimate the joint trajectories of individual robots. This makes our technique more scalable in the number of local maps, and more robust when overlap among local maps is minimal (at least one common structure). This can be observed in our results where maps built by up to 10 individual robots are merged. This can be compared to results in [43] where they merge up to 6 maps.

9.2 VR-SLAM

9.2.1 Overview

In a single robot SLAM a joint posterior over trajectories, $x_{1:t}$, and maps, m_t, is maximized constrained by a sequence of range measurements, $z_{1:t}$ and odometry readings, $u_{1:t}$. The goal is to find $\underset{x_{1:t}, m_t}{\text{argmax}}\ p(x_{1:t}, m_t | z_{1:t}, u_{1:t})$. For more details see [251]. A very successful framework for such optimization is Rao-Blackwellized particle filtering as presented in [117]. In this framework the posterior is represented using a set of particles. Each particle represents a trajectory and an associated map. At every time step t, the most likely particle is used by the robot for navigational purposes. A range measurement z_t at time t captures a small part of the environment as a local scan. An odometry reading u_t provides the update information for the robot's pose. Hence this optimization can be viewed as the process of consistently merging a sequence of local scans into a global map by tracking multiple hypotheses.

In the proposed VR-SLAM a similar optimization is performed. For ease of understanding the optimization process we imagine a *virtual* robot trying to navigate using the individual robots as its sensors. The local maps built by the individual robots replace the range measurements (local scans). The odometry readings are derived from registration of similar structures in local maps. The main differences between a single robot SLAM and our VR-SLAM are due to the differences in motion model and perception model of the virtual robot. This leads to differences in designing the *proposal distribution* and computing the *importance weights* of the particles when using the particle filtering framework. The design of the proposal is based on the motion model of the robot while computing weights is based on the perception model. Thus, once we formulate the navigational behavior of our virtual robot, the framework used in [117] can be adapted for maximizing the joint posterior of "virtual" trajectories and maps built by the virtual robot. This results in a solution to the map-merging problem. The motion model and proposal are described in Section 9.3. The perception model and importance weights are described in Section 9.4. Henceforth we use the following notation: (i) Z to denote the set of local maps to be merged, and (ii) φ_t to denote the index of the local map that is merged at time step t. Usually the individual robots can be ordered based on their proximities; this gives us the *sequence* of local maps i.e. $Z[\varphi_{1:t}]$. Even if such an ordering is not available, the framework remains similar except that the motion model distribution has a larger number of modes because at each time step structure registration is performed between merged and *all* unmerged local maps instead of just one unmerged map.

9.2.2 Rao Blackwellized Sampling Importance Resampling (RB-SIR)

- **Recursive Bayesian estimation:** Since Rao-Blackwellization with state decomposition allows us to factorize the SLAM posterior as $p(x_{1:t}, m_t|z_{1:t}, u_{1:t}) = p(m_t|x_{1:t}, z_{1:t}) \cdot p(x_{1:t}|z_{1:t}, u_{1:t})$, the proposal needs to be designed based on $p(x_{1:t}|z_{1:t}, u_{1:t})$. $p(m_t|x_{1:t}, z_{1:t})$ can be computed analytically [200]. Assuming 1st order Markov process and observational independence this is estimated recursively using Bayesian estimation as below ([65]):

$$p(x_{1:t}|z_{1:t}, u_{1:t}) = \frac{p(z_{1:t}, u_{1:t}|x_{1:t})p(x_{1:t})}{p(z_{1:t}, u_{1:t})} \text{ (using Bayes theorem)}$$

$$= \frac{p(z_t, u_t|z_{1:t-1}, u_{1:t-1}, x_{1:t})\, p(z_{1:t-1}, u_{1:t-1}|x_{1:t-1})\, p(x_t|x_{1:t-1})\, p(x_{1:t-1})}{p(z_t, u_t|z_{1:t-1}, u_{1:t-1})\, p(z_{1:t-1}, u_{1:t-1})}$$

$$= p(x_{1:t-1}|z_{1:t-1}, u_{1:t-1}) \frac{p(z_t, u_t|z_{1:t-1}, u_{1:t-1}, x_{1:t-1})\, p(x_t|x_{t-1})}{p(z_t, u_t|z_{1:t-1}, u_{1:t-1})}$$

  Once we have the pdf we can estimate any function over the state vector as:

$$E[f(x_{1:t})] = \int f(x_{1:t})p(x_{1:t}|z_{1:t}, u_{1:t})dx_{1:t} \tag{9.1}$$

  The posterior density can be non-Gaussian undergoing non-linear dynamics and hence Kalman filtering ([139]) is not sufficiently powerful for this case. Hence filtering is typically done by simulating the posterior density using randomly drawn samples from the density. This can be done either iteratively [186] or non-iteratively using Sampling Importance Resampling (SIR) algorithm [224, 223]. [108, 237, 103] provide a nice overview of such simulation perspective and algorithms. We focus on the SIR algorithm as this is non-iterative and computationally efficient and easy to implement.

- **Importance Sampling/Resampling:** In simulation based filtering, posterior density is represented using N independently identically drawn samples and any functional estimate of the state vector can be made using discrete approximation as shown below:

$$E[f(x_{1:t})] \approx \frac{1}{N}\sum_{i=1}^{N} f(x_{1:t}^{(i)}), \text{ where } x_{1:t}^{(i)} \sim p(x_{1:t}|z_{1:t}, u_{1:t}) \tag{9.2}$$

  The error of approximation drops at the rate of $\frac{1}{\sqrt{N}}$ as $N \to \infty$ [199]. In practice it is very hard to truly sample from $p(x_{1:t}|z_{1:t}, u_{1:t})$ so an easier to sample posterior $\pi(x_{1:t}|z_{1:t}, u_{1:t})$ is used and the samples are given importance weights (or probability masses). This pdf is often called *proposal* distribution or importance distribution. The weights are derived below as shown in

[258]:

$$
\begin{aligned}
E[f(x_{1:t})] &= \int f(x_{1:t}) p(x_{1:t}|z_{1:t}, u_{1:t}) dx_{1:t} \text{ (from Eq.(9.1))} \\
&= \int f(x_{1:t}) \frac{p(x_{1:t}|z_{1:t}, u_{1:t}) \pi(x_{1:t}|z_{1:t}, u_{1:t})}{\pi(x_{1:t}|z_{1:t}, u_{1:t})} dx_{1:t} \\
&= \int f(x_{1:t}) \frac{\boxed{p(z_{1:t}, u_{1:t}|x_{1:t}).p(x_{1:t})} \pi(x_{1:t}|z_{1:t}, u_{1:t})}{p(z_{1:t}, u_{1:t}) \boxed{\pi(x_{1:t}|z_{1:t}, u_{1:t})}} dx_{1:t} \text{ (using Bayes theorem)} \\
&= \frac{1}{p(z_{1:t}, u_{1:t})} \int f(x_{1:t}) \boxed{w_t(x_{1:t})} \pi(x_{1:t}|z_{1:t}, u_{1:t}) dx_{1:t} \text{ , } \left(\text{where } w_t(x_{1:t}) = \frac{p(z_{1:t}, u_{1:t}|x_{1:t}) p(x_{1:t})}{\pi(x_{1:t}|z_{1:t}, u_{1:t})}\right) \\
&= \frac{\int f(x_{1:t}) w_t(x_{1:t}) \pi(x_{1:t}|z_{1:t}, u_{1:t}) dx_{1:t}}{\int p(z_{1:t}, u_{1:t}|x_{1:t}).p(x_{1:t}) dx_{1:t}} \\
&= \frac{\int f(x_{1:t}) w_t(x_{1:t}) \pi(x_{1:t}|z_{1:t}, u_{1:t}) dx_{1:t}}{\int \boxed{p(z_{1:t}|x_{1:t}).p(x_{1:t})} \frac{\pi(x_{1:t}|z_{1:t}, u_{1:t})}{\boxed{\pi(x_{1:t}|z_{1:t}, u_{1:t})}} dx_{1:t}} \\
&= \frac{\int f(x_{1:t}) w_t(x_{1:t}) \pi(x_{1:t}|z_{1:t}, u_{1:t}) dx_{1:t}}{\int \boxed{w_t(x_{1:t})} \pi(x_{1:t}|z_{1:t}, u_{1:t}) dx_{1:t}} \\
&\approx \frac{\frac{1}{N} \sum_{i=1}^{N} f(x_{1:t}^{(i)}) w_t(x_{1:t}^{(i)})}{\frac{1}{N} \sum_{i=1}^{N} w_t(x_{1:t}^{(i)})} = \frac{1}{N} \sum_{i=1}^{N} \tilde{w}_t(x_{1:t}^{(i)}) f(x_{1:t}^{(i)}), \text{ where } x_{1:t}^{(i)} \sim \pi(x_{1:t}|z_{1:t}, u_{1:t})
\end{aligned}
$$

$$
\text{and } \tilde{w}_t(x_{1:t}^{(i)}) = \frac{w_t(x_{1:t}^{(i)})}{\sum_{i=1}^{N} w_t(x_{1:t}^{(i)})} \text{ (normalized importance weights)}
$$

Using the state space assumptions (1st order Markov and observational independence given state) the weights can be recursively estimated as follows:

$$
\begin{aligned}
w_t &= \frac{p(z_{1:t}, u_{1:t}|x_{1:t}) p(x_{1:t})}{\pi(x_{1:t}|z_{1:t}, u_{1:t})} \\
&= \frac{p(z_t, u_t|z_{1:t-1}, u_{1:t-1}, x_{1:t}) \boxed{p(z_{1:t-1}, u_{1:t-1}|x_{1:t-1})} p(x_t|x_{1:t-1}) \boxed{p(x_{1:t-1})}}{\pi(x_t|x_{1:t-1}, z_{1:t}, u_{1:t}) \boxed{\pi(x_{1:t-1}|z_{1:t-1}, u_{1:t-1})}} \\
&= \boxed{w_{t-1}} \frac{p(z_t, u_t|z_{1:t-1}, u_{1:t-1}, x_{1:t}) p(x_t|x_{1:t-1})}{\pi(x_t|x_{1:t-1}, z_{1:t}, u_{1:t})}
\end{aligned}
$$

The variance of the importance weights plays a crucial role in how accurately the samples represent the posterior [62]. Ideally all the weights must be equal making the variance to be zero and the samples to be randomly distributed according to $p(x_{1:t}|z_{1:t}, u_{1:t})$. But [145] shows that the variance of the importance weights increases stochastically over time. This is called *weight degeneracy* problem. Resampling helps in discarding the low weighted particles but does not really solve the degeneracy problem. Also if resampling is done when weight distribution has high variance, the "fitter" particles get multiple copies and after a few iterations the particles collapse to a single particle. Several strategies exist that use Markov chain transition kernels $\mathcal{K}(x_{1:t}|\tilde{x}_{1:t})$ such that $\int \mathcal{K}(x_{1:t}|\tilde{x}_{1:t}) p(\tilde{x}_{1:t}|z_{1:t}, u_{1:t}) = p(x_{1:t}|z_{1:t}, u_{1:t})$ to move samples drawn from $p(\tilde{x}_{1:t}|z_{1:t}, u_{1:t})$ to regions where variance is low [9, 42, 187]. [110] allows for particles to move from one subspace to another subspace by allowing reversible moves while [8] allows even moves along different dimensions. These clever MCMC moves "spread" the particles to reduce variance but do not explore low variance regions with high likelihood regions that are possible with better proposals. Hence the choice of π plays a very crucial role in the effectiveness and accuracy of the particle filter and is discussed below.

- **Proposal distribution:** $\pi(x_t|z_{1:t}, u_{1:t})$ has to be chosen in such a way that the samples drawn are around the *likelihood* ($p(z_{1:t}, u_{1:t}|x_{1:t})$) so that the weights are similar and the variance stays low. In fact the optimal π, as introduced in [283] and proved in [62], that minimizes the variance of the weights is given by

$$p(x_t|x_{1:t-1}, z_{1:t}, u_{1:t}) = \frac{p(z_t, u_t|z_{1:t-1}, u_{1:t-1}, x_{1:t})p(x_t|x_{t-1})}{p(z_t, u_t|z_{1:t-1}, u_{1:t-1}, x_{1:t-1})} \tag{9.3}$$

This proposal takes into account the simulated trajectory $x_{1:t-1}$ and the observations (including the latest) $z_{1:t}$. Using this optimal proposal Eq. becomes:

$$\begin{aligned} w_t &= w_{t-1}\frac{\cancel{p(z_t, u_t|z_{1:t-1}, u_{1:t-1}, x_{1:t})}\cancel{p(x_t|x_{1:t-1})}}{\frac{\cancel{p(z_t,u_t|z_{1:t-1},u_{1:t-1},x_{1:t})}\cancel{p(x_t|x_{t-1})}}{p(z_t,u_t|z_{1:t-1},u_{1:t-1},x_{1:t-1})}} \\ &= w_{t-1}p(z_t, u_t|z_{1:t-1}, u_{1:t-1}, x_{1:t-1}) \\ &= w_{t-1}\int p(z_t, u_t|z_{1:t-1}, u_{1:t-1}, x_{1:t})p(x_t|x_{t-1})dx_t \end{aligned}$$

Unfortunately sampling from $p(x_t|x_{1:t-1}, z_{1:t}, u_{1:t})$ exactly is computationally very expensive ([65]) and hence several deterministic approximations have been proposed to implement the optimal importance function [258, 63]. There are techniques like "prior editing" ([108]), "rejection methods" ([245]) which are based on heuristics of sampling particles from high likelihood regions but are computationally expensive and not suitable for online applications. [210, 258, 117] all propose different techniques towards approximating the optimal proposal Eq. (9.3). The basic underlying idea is to use Gaussian approximation of Eq. (9.3) using *prior* combined with the latest observation as

$$p(x_t|x_{1:t-1}, z_{1:t}, u_{1:t}) \approx p_{\mathcal{G}}(x_t|x_{1:t-1}, u_t, z_t) = \mathcal{N}(\hat{x_t}, cov[x_t]) \tag{9.4}$$

[210] uses two stage sampling using "auxiliary" variables to sample in higher dimension to be robust to outliers, [258] uses "unscented transforms" ([137, 138]) on "sigma-points" of the Gaussian to achieve robustness in highly non-linear processes. [197, 118, 117] implement this proposal in the SLAM context using importance sampling and Gaussian approximation as

below:

$$p(x_t|x_{1:t-1}, z_{1:t}, u_{1:t}) = p(x_t| x_{1:t-2}, x_{t-1}, z_{1:t-1}, z_t, u_{1:t-1}, u_t)$$

$$= p(x_t| m_{t-1}, x_{t-1}, z_t, u_t), \ (\because \text{ conditioning on } x_{1:t-2}, z_{1:t-1}, u_{1:t-1} \text{ is equivalent to conditioning on } m_{t-1})$$

$$= \frac{p(z_t|m_{t-1}, x_t)p(x_t|x_{t-1}, u_t)}{p(z_t|m_{t-1}, x_{t-1}, u_t)}$$

$$= \frac{p(z_t|m_{t-1}, x_t)\, p(x_t|x_{t-1}, u_t)}{\int p(z_t|m_{t-1}, x_t)\, p(x_t|x_{t-1}, u_t)dx_t}$$

$$= \frac{l(x_t)\, p(x_t|x_{t-1}, u_t)}{\int l(x_t)\, p(x_t|x_{t-1}, u_t)dx_t}, \ \text{(where } l(x_t) = p(z_t|m_{t-1}, x_t) \text{ is computed using an appropriate sensor model [25}$$

Now the above expression is approximated using a set of samples $\left\{x_t^{(i)}\right\}_{i=1}^{K}$ with importance weights. The set of samples in [118] are drawn from $p(x_t|x_{t-1}, u_t)$ such that $l(x_t^{(i)}) > k$ and the importance weights are given by $\frac{l(x_t^{(i)})}{\frac{1}{K}\sum_{i=1}^{K} l(x_t^{(i)})}$. In [117] the samples are drawn around $\text{argmax}_x p(x|m_{t-1}, z_t, x_{t-1} \oplus u_t)$. The local maximum search can be performed using any standard scan-matching technique while they use VASCO a scan-matching module in CARMEN [196, 195]. The importance weights in their case are given by $\frac{l(x_t^{(i)})p(x_t^{(i)}|x_{t-1},u_t)}{\sum_{i=1}^{K} l(x_t^{(i)})p(x_t^{(i)}|x_{t-1},u_t)}$. Once they obtain the discrete approximation they fit a Gaussian kernel $\mathcal{N}(\hat{x}_t, cov[x_t])$ for efficient sampling. [197] computes $l(x_t)$ using landmarks while [118, 117] improves upon them using grid-based scan-matching and/or motion model of the robot to compute $l(x_t)$. With this approximation of the proposal the recursive weight update is done as follows:

$$w_t = w_{t-1} \int p(z_t, u_t|z_{1:t-1}, u_{1:t-1}, x_{1:t})p(x_t|x_{t-1})dx_t$$

$$= w_{t-1} \int p(z_t, u_t|m_{t-1}, x_{t-1:t})p(x_t|x_{t-1})dx_t$$

$$= w_{t-1} \int p(z_t|m_{t-1}, x_{t-1}, x_t, u_t)\, p(u_t|m_{t-1}, x_{t-1}, x_t)\, p(x_t|x_{t-1})dx_t$$

$$= w_{t-1} \int p(z_t|m_{t-1}, x_t) \frac{p(u_t|m_{t-1},x_{t-1})p(x_t|x_{t-1},u_t,m_{t-1})}{p(x_t|x_{t-1},m_{t-1})} p(x_t|x_{t-1})dx_t \ \text{(using Bayes theorem)}$$

$$= w_{t-1}c \int p(z_t|m_{t-1}, x_t)\frac{p(x_t|x_{t-1}, u_t)}{\cancel{p(x_t|x_{t-1})}}\cancel{p(x_t|x_{t-1})}dx_t \ (\because \ x_t \text{ is independent of } m_{t-1})$$

The above integral is approximated by a sum over discrete set of samples $\left\{x_t^{(i)}\right\}_{i=1}^{K}$ that was used for approximating the proposal. Hence in [118] the recursion becomes $w_t = w_{t-1}\frac{c}{K}\sum_{i=1}^{K} p(z_t|m_{t-1}, x_t^{(i)})$ while in [117] the recursion would be $w_t = w_{t-1}c\sum_{i=1}^{K} p(z_t|m_{t-1}, x_t^{(i)})p(x_t^{(i)}|x_{t-1}, u_t)$. The main idea in choosing the set is that the set of x_t around the maximum likelihood determines the value of the integral. The other values of x_t do not influence significantly. This can be seen in Fig. 9.1. Hence for finite and efficient discrete approximation of the integral it is important to choose x_t around such regions. [70] follows a different strategy by which it tries to approximate the sum by drawing large number of x_t and using efficient non-trivial data-structures for updates. As mentioned in [117] it is important that to sample x_t separately if there are multiple modes as shown in the Fig. 9.2. This will increase the number of particles depending on how many modes $p(z_t|m_{t-1}, u_t)$ has since the filter should "track" all those modes. While in the case of standard single robot SLAM multiple modes do not occur often in practice, in the scenario of our V-SLAM it often happens that there are multiple peaked likelihoods. This happens because there could be several individual robots whose maps can be "visited" to be merged with equally good chance. We differ from [117] in the way the peaked regions are found. They use an initial estimate $x'_t = x_{t-1} \oplus u_{t-1}$ and then find a local maximum using a hill-climbing strategy guided by some kind of scan-correlation [251]. We find the peaked likelihood regions by searching in a discrete space of poses obtained using structure based motion control as explained in section 9.3. The search is guided by enhanced scan-correlation metric (section 9.4) based on the metric introduced in [24] which is very similar to popular "Grassfire transform" in computer vision literature. As discussed in [117] we also employ adaptive resampling using the metric $N_{eff} = \frac{1}{\sum_{i=1}^{N}(\tilde{w}^{(i)})^2}$ as formulated in [64]. When we resample we use "residual resampling" technique [61] and retain only N samples of $x_{1:t}$. So effectively the set of samples varies as the algorithm progresses. If there are multiple modes and the variance is low (as indicated by $N_{eff} \geq N/2$) more number of samples are kept. Otherwise resampling is done to bring the number of samples down to N.

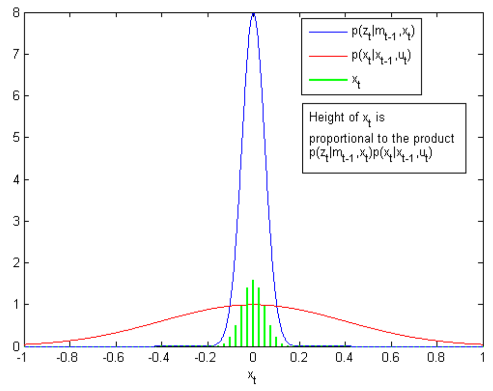


Figure 9.1: Discrete approximation of the integral: The samples with longer green lines indicate the important samples that influence the accuracy of approximation.

9.3 Motion from structure registration

We use the optimal proposal with ability to handle multiple modes as described in [55] which is an improvement over [117]. There are two key differences between implementing the proposal for regular SLAM and that for VR-SLAM: (1) Drawing the follower poses from the distribution modeling

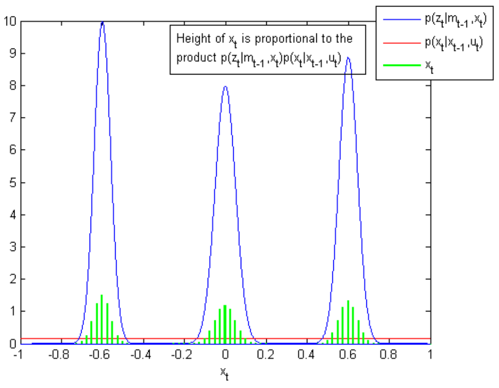


Figure 9.2: $p(z_t|m_{t-1}, u_t)$ having multiple modes and $p(x_t|x_{t-1}, u_t)$ being insignificant and noisy. In such case the samples have to be drawn separately from the multiple modes which increases the number of particles.

the motion of the robot i.e. $p(x_t|x_{t-1}, u_t)$, which is explained in this section. (2) The computation of the likelihood of poses i.e. $p(z_t|x_j, m_{t-1})$ (Section 9.4). The rest of the procedure remains same as described in [55]. Simulation from the above distribution is based on the motion model of the robot. Now we explain the structure registration based motion model for our virtual robot. The virtual robot always moves in the global frame. Its pose is initialized at the origin of one of the local maps, which becomes the reference coordinate frame (global frame). Its motion update is then guided by "structure registration" among local maps. Its position in a local map is always at the local origin. Similar structures among local maps are extracted and registered using closed form solutions like in [126, 125, 11]. The correspondences are obtained by shape matching between similar structures. u_t encapsulates the motion updates by registering similar structures between local maps $Z[\varphi_{t-1}]$ and $Z[\varphi_t]$. There could be several similar structures between $Z[\varphi_{t-1}]$ and $Z[\varphi_t]$. As a consequence, our proposal distribution is multi-modal with peaks around the poses predicted using structure registration. Distribution of typical, odometry based motion model is shown in Fig. 9.3 (a) while that in our case is shown in Fig. 9.3 (b). Fig. 9.4 shows a sample motion update process with two modes. We note that each possible pose update is actually a transformation of the local map into the global frame of m_{t-1}.

9.4 Perception model using image similarity

The main component in recursively estimating importance weights as described in [117] is $p(z_t|x_j, m_{t-1})$, which is also used in implementing the optimal proposal [117]. This typically involves "scan matching".

Scan-matching is the problem of finding a rigid transformation such that the overlapping

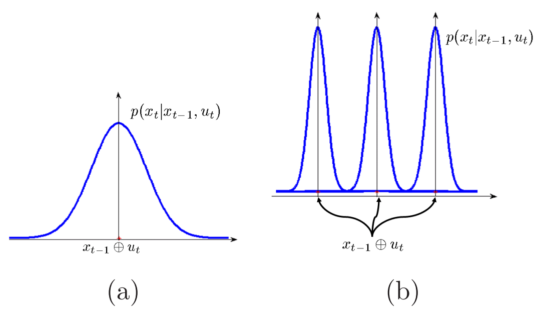


Figure 9.3: (a) Typical distribution of odometry based motion model in a single robot SLAM. (b) In our case the distribution is multi-modal with number of modes being equal to the number of pairs of similar structures.

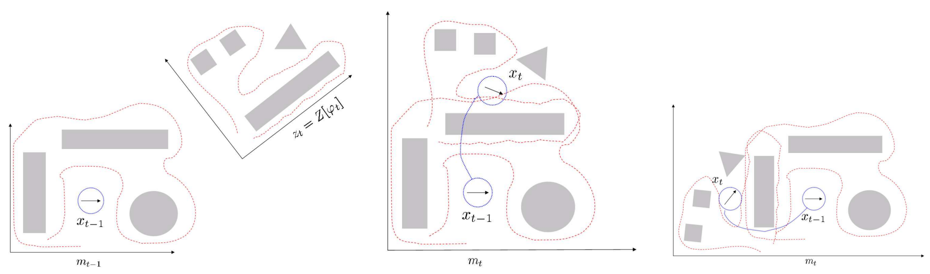


Figure 9.4: Left: The virtual robot's pose in the global frame is shown with an arrow inside the circle. The new local map, $Z[\varphi_t]$ is shown in its local frame. The dotted red lines are the trajectories of the individual robots. Middle: One possible structure registration updates the pose of the virtual robot. The virtual robot's jump is shown in dotted blue line. Right: Another possible structure registration leads to a different pose update for the virtual robot.

region of a scan is registered onto the corresponding part of a reference scan. This makes the problem similar to that of image registration restricted to rigid transformations. The main difficulty of course is to find the corresponding regions. In a way this is like map-merging of maps built by two robots. In fact this reveals an intriguing insight that all the various techniques present in the vast fruitful literature of robot mapping and localization address the *scalability* of the solutions to the underlying "correspondence/data association" problem. This also makes it clear why VR-SLAM is a highly scalable solution compared to other *pair-wise* map-merging techniques. The problem of estimating the true correspondences given two maps is NP-complete since the combinatorial optimization problem of finding "Maximum Common Subgraph", which is NP-complete [101] can be reduced to this problem. In practice however because of the type of graphs several heuristic based approximation algorithms have been applied in several domain specific applications like [267, 39, 203, 119]. In the context of map-merging the approximation algorithms usually work well [15] but the main issue is that the graphs themselves are *noisy*. Hence probabilistic approaches are typically used to address the "confidence" of the solutions. Typically computationally efficient Bayesian solutions are possible under the Markovian assumptions and observational independence.

The computation of $p(z_t|x_j, m_{t-1})$ is based on the perception model of the robot. As mentioned earlier the local maps built by the individual robots form the sensor readings of the virtual robot. Hence z_t is characterized by the occupancy grid of the local map $Z[\varphi_t]$. Our approach is motivated by correlation based models used in regular SLAM [251]. The standard correlation metric based on the normalized quadratic distance does not address the "see through walls" problem [251]. A correlation metric Δ based on image distance ψ (introduced in [24]) and a heuristic to address the inconsistent merge issue was introduced in [23]. The heuristic is based on the idea of "locking" the merge process to avoid slipping into inconsistent merges with small ψ. This heuristic although *preventing* slipping into inconsistent merges, fails to *guide* the process. This is because their random walks are guided by the gradients computed on ψ.

We use image similarity measures (motivated by those used for image registration as in [23]) to rank the transformations. The occupancy grids are converted to digital images by discretizing the cell probability values into a set $C = \{$*free*, *occupied* and *unknown*$\}$. Each cell in the occupancy grid becomes a pixel in the converted image.

There are two main components of our measure ρ viz. Ψ and γ. Ψ accounts for measuring the "overlap" between the maps and γ accounts for "see through walls" issue that needs to be addressed in the correlation-based modeling of sensor information for robot mapping. We first describe Ψ and then γ in the following text.

$$\Psi(m_1, m_2) = \sum_{c \in \mathcal{C}} s(m_1, m_2, c, c) + s(m_2, m_1, c, c) \tag{9.5}$$

where

- $\mathcal{C} = C \setminus \{unknown\}$
- $s(m_1, m_2, c_1, c_2) = \frac{\sum_{m_1[p_1]=c_1} e^{\frac{x(2\mu - x)}{2\sigma^2}}}{\#_{c_1}(m_1)}$.
- $x = min\,\{md(p_1, p_2)|m_2[p_2] = c_2\}$.
- $m_1[p_1]$ denotes the value c of map m_1 at position $p = (x, y)$.
- $md(p_1, p_2) = |x_1 - x_2| + |y_1 - y_2|$ is the Manhattan-Distance between the two points p_1 and p_2.
- $\#_{c_1}(m_1) = \#p_1|m_1[p_1] = c_1$ is the number of cell in m_1 with value c.
- μ and σ are the parameters that influence the Gaussian kernel. Gaussian kernel is applied because the sensor noise is typically Gaussian.

Ψ differs from ψ of [23] in that we compute *similarity* of the images and that we do it in a way similar to using likelihood fields of maps [251]. The basic idea is to treat the images as arrays of pixels and reward matching pixels based on their values and relative positions. The arrays are scanned for each of the class of pixels. For each pixel of a value the distance to closest similar valued pixel in the other image is computed and its score proportional to the Gaussian of the distance. Fortunately the distances can be computed in linear time in the number of pixels in the images using "distance-maps" [23].

The component γ is defined as:

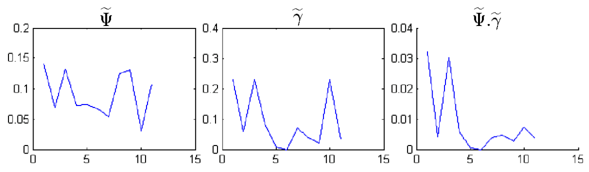
$$\gamma(m_1, m_2) = -\log(d(m_1, m_2)) - d(m_1, m_2) \tag{9.6}$$

where $d(m_1, m_2) = \sum_{c_1 \neq c_2 | \{c_1, c_2\} \subset \mathcal{C}} s(m_1, m_2, c_1, c_2) + s(m_2, m_1, c_2, c_1)$. γ is designed such that its magnitude is proportional to the inconsistency of a merge. Inconsistency is defined as the mismatch in the perception of the robot from two different positions. The mismatch in the perception is based on the number of disagreeing pixels in the two images. Thus its computation involves similar steps as those for Ψ. While Ψ accumulates score based on the matching pixels $c_1 = c_2$, γ accumulates score based on $c_1 \neq c_2$.

$\forall\ t_i \in T$ we compute $\Psi_i = \Psi(m_1, t_i(m_2))$ and $\gamma_i = \gamma(m_1, t_i(m_2))$ as:

$$\tilde{\Psi}_i = \frac{\Psi_i}{\sum_{i=1}^{n} \Psi_i}$$
$$\tilde{\gamma}_i = \frac{\gamma_i - min(\gamma_i)}{\sum_{i=1}^{n}(\gamma_i - min(\gamma_i))}$$

The $\{\tilde{\Psi}_i\}_{i=1}^{n}$ and $\{\tilde{\gamma}_i\}_{i=1}^{n}$ form discrete *independent* probability distributions for the likelihood of transformations and hence the likelihood score $p(z_t|m_{t-1}, x_j)$ is given by the product $\widetilde{\Psi}_j.\widetilde{\gamma}_j$. The likelihood scores of a typical set of poses is shown in Fig. 9.5. The figure also shows how neither $\widetilde{\Psi}$ nor $\widetilde{\gamma}$ by itself is sufficient but their combination helps improve the likelihood scores.



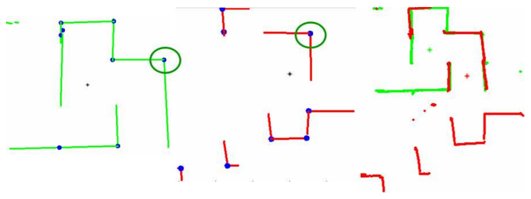


Figure 9.5: The scores for a set of 11 poses are shown. The $8^{th}, 9^{th}$ and 11^{th} poses have high $\tilde{\Psi}$ indicating good overlap but low consistency scores as seen by $\widetilde{\gamma}$. The final likelihood scores obtained by their product reflect scores in which both overlap and consistency are satisfied. Corners extracted from two local maps (green and red) are shown above in blue. A pair of similar corners are marked with green circles. The merged map obtained by registering the two similar corners is shown in the right.

9.5 Experimental results

We applied our technique to the data collected by National Institute of Standards and Technology (NIST) in a "maze" type of environment[1]. We are provided only with the local maps without any initial estimate of the relative poses. For feature extraction we fit lines to the data points and determine corners. For typical indoor local maps, corners are sufficiently distinct feature descriptors to represent the environment structure. Sample corners extracted from two partial maps can be seen in Fig. 9.5. On average there were about 10 corners in a local map and there were about 5 matching corner pairs between two local maps. The results are shown for merging maps

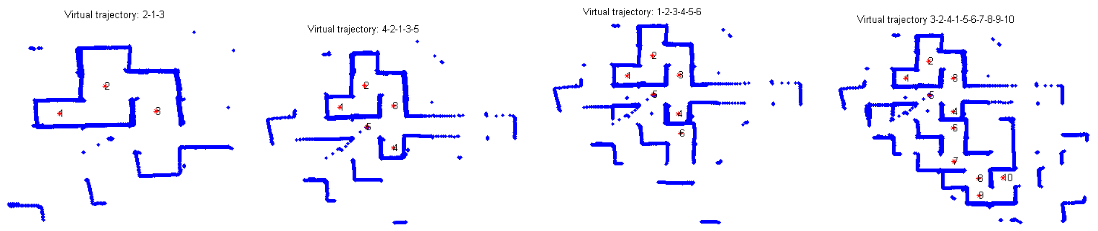

Virtual trajectory: 2-1-3
Virtual trajectory: 4-2-1-3-5
Virtual trajectory: 1-2-3-4-5-6
Virtual trajectory 3-2-4-1-5-6-7-8-9-10
1
2
3
4
5
6
7
8
9
10

Figure 9.6: Merged maps of $3, 5, 6$ and 10 robots. The map indices are placed at the local origins of the individual maps. The sequence of the maps merged is shown in the respective titles.

from $3, 5, 6$ and 10 robots in Fig. 9.6. We used an average of 50 to 100 particles. As can be seen we could successfully merge maps of 10 robots. Since we did not assume the sequence of local maps our process estimates the sequence also automatically as mentioned in Section 9.2.1. For e.g. the optimal sequence of maps for 10 robots (Fig. 9.6) is $(3, 2, 4, 1, 5, 6, 7, 8, 9, 10)$. The time complexity of our algorithm depends on the number of local maps to be merged (N), the average number of corners in each map (c) and the number of particles (N_p). If the sequence of local maps to be merged $(Z[\varphi_{1:t}])$ is known then the complexity is $O(N_p c^2 N^2)$ otherwise it is $O(N_p c^2 N^3)$.

[1]We would like to thank Raj Madhavan (NIST) for providing the dataset.

CHAPTER 10

Performance Study of $6D$ LuM and FFS SLAM

The focus of this chapter is on the performance comparison of two simultaneous localization and mapping (SLAM) algorithms namely $6D$ Lu/Milios SLAM and Force Field Simulation (FFS). The two algorithms are applied to a $2D$ data set. Although the algorithms generate overall visually comparable results, they show strengths & weaknesses in different regions of the generated global maps. The question we address in this paper is, if different ways of evaluating the performance of SLAM algorithms project different strengths and how can the evaluations be useful in selecting an algorithm. We will compare the performance of the algorithms in different ways, using grid and pose based quality measures.

10.1 Introduction

The simultaneous localization and mapping (SLAM) problem is one of the basic problems in autonomous robot navigation. In the past many solutions of the simultaneous localization and mapping (SLAM) problem have been proposed [250]. However, it is difficult for engineers and developers to choose a suitable algorithm, due to a lack of true benchmarking experiments. In the well-known Radish (The Robotics Data Set Repository) repository [128] algorithms and results as bitmapped figures are available, but the algorithms have not been compared against each other. A valuable source for state of the art performance are competitions like RoboCup [75], Grand Challenge [46] or the European Land Robotics Trial [89]. However, the aim of such competitions is to evaluate whole systems under operational conditions, but are not well suited for the performance evaluation of vital components like perception. This paper presents two methodologies for comparing the results of state of the art SLAM algorithms, namely $6D$ LuM [32] and FFS [161].

LuM and FFS SLAM, treat the mapping problem as an optimization problem, that is a maximal likelihood map learning method. The algorithms seek to find a configuration ξ^*, i.e., scan poses that maximizes the likelihood of observations and could be written as

$$\xi^* = \underset{\xi}{\operatorname{argmax}} F(\xi),$$

where F is a function measuring the map quality or likelihood.

This paper is organized as follows: After an overview of related work, section 10.3 will give a brief description of the compared SLAM algorithms. Section 10.4 presents our evaluation methodology, followed by the results. Section VI concludes.

10.2 Related Work

10.2.1 Robot Mapping

State of the art for metric maps are probabilistic methods, where the robot has probabilistic motion models and perception models. Through integration of these two distributions with a Bayes filter, e.g., Kalman or particle filter, it is possible to localize the robot. Mapping is often an extension to this estimation problem. Besides the robot pose, positions of landmarks are also estimated. Closed loops, i.e., revisiting a previously visited area of the environment, play a special role here: Once detected, they enable the algorithms to bound the error by deforming the mapped area to yield a topologically consistent model. For e.g. [240] addresses the issues in loop-closing problems. Several strategies exist for solving SLAM. Thrun [250] surveys existing techniques, like, maximum likelihood estimation, Expectation Maximization, Extended Kalman Filter, Sparsely Extended Information Filter SLAM. FastSLAM [252] and its improved variants like [117] use Rao-Blackwellized particle filters.

SLAM in well-defined, planar indoor environments is considered solved. However, little effort has been spent in *comparing* the performance evaluation of the SLAM algorithms. Given vast literature and various successful approaches for SLAM, such comparative studies are needed to choose appropriate SLAM algorithms for specific applications.

10.2.2 Performance Evaluation

Most research in the SLAM community aims at creating consistent maps. Recently, on the theoretical side of SLAM, Bailey et al. proves that EKF-SLAM fails in large environments [16] and FastSLAM is inconsistent as a statistical filter: it always underestimates its own error in the medium to long-term [17] that is it becomes over-confident. Besides focusing on such consistency issues, little effort has been made in *comparative* studies of SLAM algorithms.

Comparing two or more SLAM algorithms needs quantitative performance metrics like robustness, rate of convergence, quality of the results. Though the metrics used for comparison in this paper are not completely new, the use of them in this context has not been done before, to the best of our knowledge. In this paper we mainly focus on the rate of convergence and quality of results of the two algorithms. They are measured in two different ways: occupancy grid based and pose based, as described in section 10.4.

10.3 Description of Mapping Algorithms

10.3.1 FFS

FFS [161] treats map alignment as an optimization problem. Single scans, possibly gained from different robots, are kept separately but are superimposed after translation and rotation to build a global map. The task is to find the optimal rotation and translation of each scan to minimize a cost function defined on this map. FFS is a gradient descent algorithm, motivated by the dynamics of rigid bodies in a force field. In analogy to Physics, the data points are seen as masses, data points of a single scan are rigidly connected with massless rods. The superimposition of scans defines the location of masses, which induces a force field. In each iteration, FFS transforms (rotates/translates) all single scans simultaneously in direction of the gradient defined by the force field under the constraints of rigid movement; the global map converges towards a minimum of the overlying potential function, which is the cost function. FFS is motivated by physics, but is adapted to the application of map alignment. It differs in the definition of the potential function, and in the choice of the step width of the gradient descent. The potential is defined as

$$P = \frac{1}{2} \sum_{p_i \in \mathcal{P}} \sum_{p_j \in \mathcal{P} \setminus p_i} \int_{\infty}^{r} \frac{m_1 m_2 \cos(\angle(p_i, p_j))}{\sigma_t \sqrt{2\pi}} e^{\left(-\frac{z^2}{2\sigma_t^2}\right)} dz \tag{10.1}$$

with $r = \sqrt{(X-x)^2 + (Y-y)^2}$,$p_i = (X, Y)$,$p_j = (x, y) \in \mathcal{P}$, $\mathcal{P}$ is the set of all transformed data points.

The potential function measures the probability of visual correspondence between all pairs of data points based on distance, direction and visual importance of data points: in (10.1) m_1, m_2 denote the visual importance of two data points, $\angle(p_i, p_j)$ the difference of direction of two points. Defining the visual importance of points dynamically is a simple interface to incorporate low or mid level perceptual properties (e.g. shape properties) into the of the global map into the optimization process. In contrast to algorithms like ICP, FFS does not work on optimization of nearest neighbor correspondences only, but (theoretically) takes into account all pairs of correspondences. Different techniques built into FFS drastically reduce the computational complexity.

FFS is steered by two parameters, σ_t in eq. 10.1 and the step width Δ_t of the gradient descent. σ steers the influence of distance between points. Initially set to a big value to accumulate information from a large neighborhood, it linearly decreases over the iterations to focus on local properties. The step width Δ_t in the FFS gradient descent is defined by an exponentially decreasing cooling process, similar to techniques like simulated annealing. Initially set to a high value it allows for significant transformations to possibly escape local minima. Decreasing the step enables local adjustment in combination with a low σ_t.

To conclude, the basic properties of FFS are

1. Data point correspondences are not made by a hard decision, but an integral between pairs of points defines the cost function instead of hard 'nearest neighbor' correspondences
2. FFS is a gradient approach, it does not commit to an optimal solution in each iteration step
3. The iteration step towards an optimal solution is steered by a 'cooling process', that allows the system to escape local minima
4. FFS transforms all scans simultaneously thus searching in $3n$ space of configurations with n scans.
5. FFS easily incorporates structural similarity modeling human perception to emphasize/strengthen the correspondences

10.3.2 $6D$ LuM

To solve SLAM, a $6D$ graph optimization algorithm for global relaxation based on the method of Lu and Milios [184] is employed, namely Lu and Milios style SLAM (LUM). Details of the $6D$ optimization, i.e., how the matrices have to be filled, can be found in [32]:

Given a network with $n+1$ nodes $X_0, ..., X_n$ representing the poses $V_0, ..., V_n$, and the directed edges $D_{i,j}$, we aim to estimate all poses optimally to build a consistent map of the environment. For simplicity, we make the approximation that the measurement equation is linear, i.e.

$$D_{i,j} = X_i - X_j.$$

An error function is formed such that minimization results in improved pose estimations:

$$\mathbf{W} = \sum_{(i,j)} (D_{i,j} - \bar{D}_{i,j})^T C_{i,j}^{-1} (D_{i,j} - \bar{D}_{i,j}). \tag{10.2}$$

where $\bar{D}_{i,j} = D_{i,j} + \Delta D_{i,j}$ models random Gaussian noise added to the unknown exact pose $D_{i,j}$. The covariance matrices $C_{i,j}$ describing the pose relations in the network are computed based on the paired points of the ICP algorithm. The error function Eq. (10.2) has a quadratic form and is therefore solved in closed form by Cholesky decomposition in the order of $\mathcal{O}(n^3)$ for n poses ($n \ll N$). The algorithm optimizes Eq. (10.2) gradually by iterating the following five steps [32]:

I) Compute the point correspondences (n closest points) using a distance threshold (here: 20 cm) for any link (i, j) in the given graph.

II) Calculate the measurement vector $\bar{D}_{ij}$ and its covariance C_{ij}.

III) From all $\bar{D}_{ij}$ and C_{ij} form a linear system $\mathbf{GX} = \mathbf{B}$.

IV) Solve for $\mathbf{X}$

V) Update the poses and their covariances.

For this GraphSLAM algorithm the graph is computed as follows: Given initial pose estimates, we compute the the number of closest points with a distance threshold (20 cm). If there are more than 5 point pairs, a link to the the graph is added.

To summarize, the basic properties of $6D$ LuM are

1. Data point correspondences are made by a hard decision using 'nearest neighbor' point correspondences
2. $6D$ LuM computes the minimum in every iteration
3. $6D$ LuM transforms all scans simultaneously
4. This GraphSLAM approach has been extended successfully to process $3D$ scans with representation of robot poses using 6 degrees of freedom.

In this paper, we process 2D laser range scans with the $6D$ LuM algorithm, i.e., in the range data the height coordinate is set 0. In this case the the algorithms shows the behaviour of the original Lu and Milios [184] GraphSLAM method.

10.4 Evaluation

Evaluation of SLAM algorithms applied to real world data often faces the problem that ground truth information is hard to collect. For example, in settings of Search and Rescue environments, data sets are scanned, which usually have no exact underlying blue print, due to the nature of the random spatial placement of (sparse) landmarks and features. Hence map inherent qualities, like entropy of the distribution of data points, must be used to infer measures of quality that reflect their ability to map the real world. In the experiments, we will compare the performance of $6D$ LuM and FFS SLAM using a grid based and a pose based approach. Especially the grid based approach will be compared to visual inspection, which in this setting could be seen as a subjective ground truth of the performance evaluation. The reason for the choice of $6D$ LuM and FFS SLAM are the following:

- Both, $6D$ LuM and FFS SLAM, are state of the art algorithms to simultaneously process multiple scans, which is needed in settings of multi-robot mapping, which is a problem that currently has stronger focus in robot mapping.

- By visual inspection, $6D$ LuM and FFS perform, intuitively spoken, alike, although differing in details. Evaluation of the algorithms should be able to report this behavior.

It should be noted that $6D$ LuM is applied here to a $2D$ dataset, to compare it to the currently available version of the FFS algorithm, which works on $2D$ scan data only. Hence the LuM performance is only evaluated on three dimensions.

10.4.1 Occupancy Grid Based

Occupancy grids are used to represent the environment by discretizing the space into grid cells that have probabilistic occupancy values accumulated by sensor readings. They were introduced by [200] and are very popular in SLAM community. Learning occupancy grids is an essential component of the SLAM process. Once built they can be used to evaluate the likelihood of the sensor readings and also be used for guiding the exploration task as they are useful for computing the information gain of actions.

The likelihood of sensor readings is computed usually using different sensor models like beam-model, likelihood-field model or map-correlation model [251]. The information gain of actions can be computed using change in entropy of the grid. We use these basic ideas to compare the outcome of the two SLAM algorithms.

We use the beam penetration model described in [70] to compute likelihood of the sensor readings. Entropy of the grid is computed as described in [239]. Once the final map is obtained we compute the log-likelihood of *all* sensor readings with trajectory given out by the algorithm as

$$\mathcal{L}(m, \mathbf{x}_{1:n}) = \sum_{i=1}^{n} \sum_{j=1}^{K} \log(p(z_{ij})|\mathbf{x}_i, m))$$

where m is the final occupancy grid, $\mathbf{x}_{1:n}$ is the final set of poses, K is the number of sensor readings at each pose and $p(z_{ij}|\mathbf{x}_i, m)$ is computed using beam-penetration model as in [70]. The log-likelihood ranges from $-\infty$ to 0 and the higher it is the better the algorithm's output.

The entropy of the map is computed based on the common independence assumption about the grid cells. Since each grid cell in the occupancy grid is a binary random variable the entropy of $H(m)$ is computed as follows as described in [239].

$$H(m) = -\sum_{c \in m} p(c) \log p(c) + (1 - p(c)) \log(1 - p(c))$$

Since the value of $H(m)$ is not independent of the grid resolution it's important either to use same resolution or to weight the entropy of each cell with its size when comparing output from two algorithms. The lower the entropy of the map the better the outcome is. It is important to note that the entropy of the map and the likelihood-scores are not completely uncorrelated.

10.4.2 Pose Based

The occupancy grid based evaluations are very useful in the sense that they do not need ”ground truths” to compare the results. But their memory requirements are proportional to the dimensionality and size of the environment. The pose based evaluations have an advantage in terms of memory requirements but require ”ground truth” data to compare to. Here we present the technique that can be used to measure the quality of the output of SLAM algorithm assuming ground truth trajectory *is* available. The ground truth data can be obtained by surveying the environment as done in [60].

The SLAM algorithm gives out a final set of poses $\mathbf{x}_{1:n}$. Let the set of ground truth poses be $\mathbf{x}^G_{1:n}$. Since each pose in $2D$ mapping has three components viz. x, y, θ we compute the average error in each of the components. It is important that both the output of SLAM algorithm and the ground truth poses are in the same global frame. This could be done by rotating and translating the set of poses such that the first corresponding pose in each set is $(0, 0, 0)$. Once the poses are in same global frame the average error in each component is computed as:

$$E(x) = \frac{1}{n}\sum_{i=1}^{n} |x_i - x_i^G|$$
$$E(y) = \frac{1}{n}\sum_{i=1}^{n} |y_i - y_i^G|$$
$$E(\theta) = \frac{1}{n}\sum_{i=1}^{n} \cos^{-1}(\cos(\theta_i - \theta_i^G))$$

$E(\theta)$ is computed as shown above so that the difference between the orientations is always between 0 and π.

10.5 Experiments

10.5.1 The Data, Visual Inspection

Both algorithms will be evaluated based on their performance on the NIST disaster data set with the same initial set of poses, see fig. 10.1. The data set consists of 60 scans and is especially complicated to map, since the single scans have minimal overlap only, and no distinct landmarks are present in the single scans. For this data set, no reliable ground truth pose data exists. This configuration was gained by random distortion of a manually gained global map.

Six sample scans are shown in fig. 10.2. The final results of LuM and FFS respectively are shown in fig. 10.3. Visual inspection of 10.3 shows the following properties:

- The overall appearance of both approaches is equal.
- The mapping quality in different details is different: while FFS performs better in the left half,

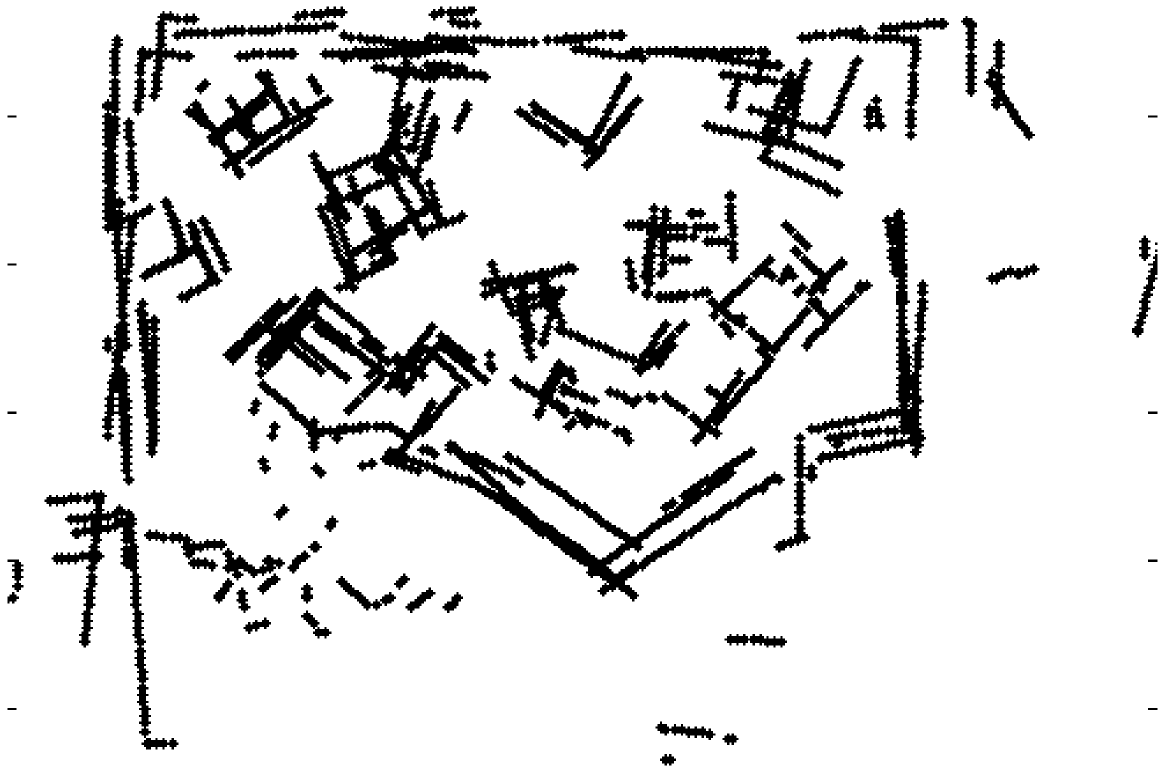


Figure 10.1: Initial configuration of the NIST data set. The data consists of 60 scans. The scale is in centimeters.

especially in the top left quarter, LuM shows a more visually consistent result in the right half, especially the top right corner.

To test if the evaluation does reflect these properties, we performed the following tests:

- First, entropy (and additional, likelihood-score) of the entire global maps (global evaluation) of both algorithms over all iterations are computed. This should reflect the behavior of both algorithms to converge towards optimal values, which should be in the same order of magnitude for both metrics.
- To check the evaluation of the different quality of mapping details in different areas, we split the result maps into four quarters and evaluated separately (regional evaluation).

In the LuM algorithm, 500 iterations were performed. FFS stopped automatically after 50 iterations, detecting a condition of changes in poses below a certain threshold. To compare all iteration steps, we extended the final result (iteration 50) to iterations $51 - 500$.

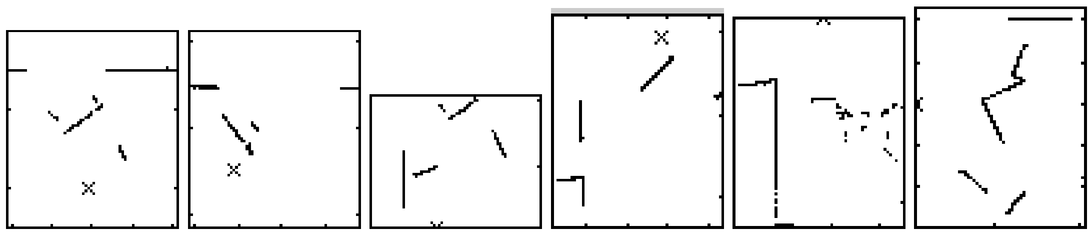


Figure 10.2: 6 example scans of the NIST data set. In fig. 10.1, they can be located on the left side.

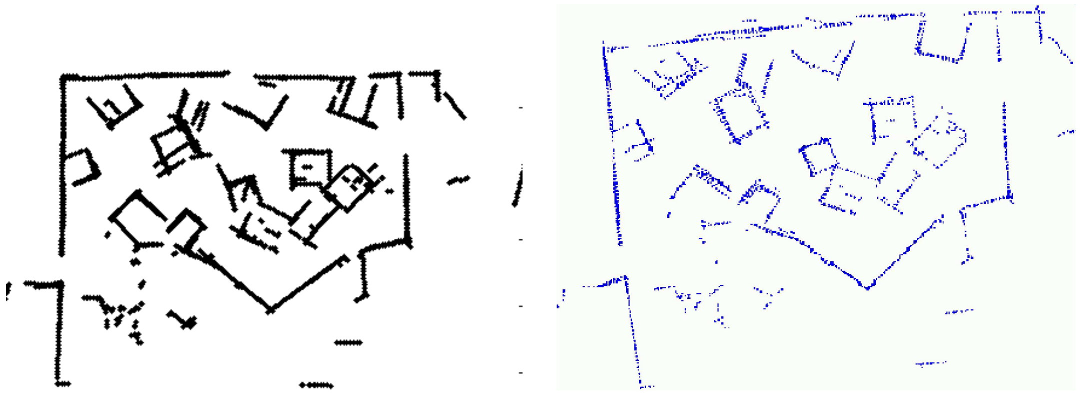


Figure 10.3: Result of FFS (left) and LuM (right) on NIST data set, initialized as in 10.1. Evaluated by the overall visual impression, both algorithms perform comparably. Differences in details can be seen especially in the top left, where FFS performs better, and the top right, where LuM is more precise.

10.5.2 Grid Based Global Evaluation

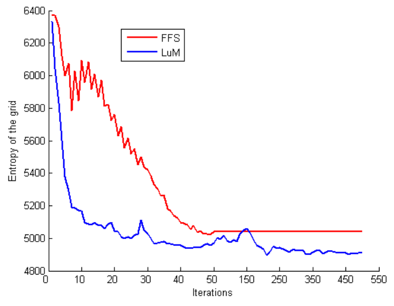
The entropies and the likelihood-scores of the maps as the algorithms progress are shown in the fig. 10.4(a) and 10.4(b) respectively.

Please note the different scale on the iteration axis in the intervals $[1-50] and (50-500]$, in the first interval the iterations increase in units of 1, whereas in the second they increase in units of 10. This holds for all following figures.

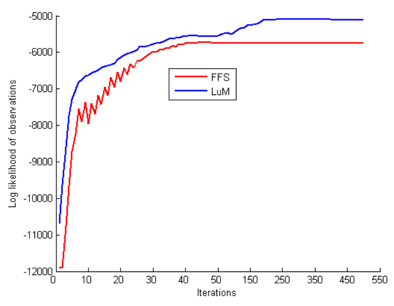
You can see that the entropy decreases non-monotonically in case of FFS while in case of LuM it tends to be monotonically decreasing. This is based in the different nature of both algorithms: FFS is gradient based approach that has a built in "cooling strategy" for the step width to possibly escape local minima. In the beginning, FFS takes bigger steps, yielding a non monotonic behavior in its target function, which is also visible in the entropy. LuM optimizes its pose in each iteration, leading to a more smooth behavior, bearing the risk of being caught in local minima. This is also reflected in the convergence behavior in terms of speed: since LuM commits to optimal solutions earlier, it converges faster in the beginning, slowing down afterwards. FFS is slower (or more positively: more careful) in the first steps, due to the choice of step width that causes a jittering behavior. After the step width is balanced, FFS reaches its optimum very quickly. Interestingly, in both cases the near optimum value is reached after about 50 iterations.

The entropy score of both algorithms is comparable, which fulfills the expectations based on the visual inspection.

Similar behavior is observed in the likelihood scores. Hence the grid based evaluation is able to reflect the properties of both algorithms in the case of global evaluation.



(a)



(b)

Figure 10.4: (a): The entropy of the map $H(m)$ at various stages of FFS and LuM. (b): The likelihood-score $\mathcal{L}(m, \mathbf{x}_{1:n})$ at various stages of the algorithms. Please note the different scale on the iteration axis in the intervals $[1 - 50] and (50 - 500]$.

10.5.3 Regional Evaluation

The maps are split into four regions, being North-West, North-East, South-West, South-East. Only the results for entropies are shown here, the likelihood scores did not lead to additional further information. We expect better results for FFS in the North-West region, whereas LuM should outperform FFS in the North-East region, results for the southern regions should not vastly differ from each other.

The results are presented in fig. 10.5. fig. 10.5(a) shows the behavior for the North-West region of the map while 10.5(b) shows for North-East, 10.5(c) for South-West and 10.5(d) for South-East.

In accord with visual inspection, FFS is evaluated to perform better on the North-West region (fig. 10.5(a)) while LuM performs better in other regions. However, looking at the difference in final values, we can see that they always differ in ranges between $\sim$ 30 and $\sim$ 80 units: (a) $\sim 430 - 480$, (b) $\sim 3200 - 3280$, (c) $\sim 278 - 309$, (d) $\sim 950 - 1000$. Hence, although the tendency in the north regions is correct, the comparison to the southern regions, which should yield a smaller distance in values, does not clearly verify the correct estimation.

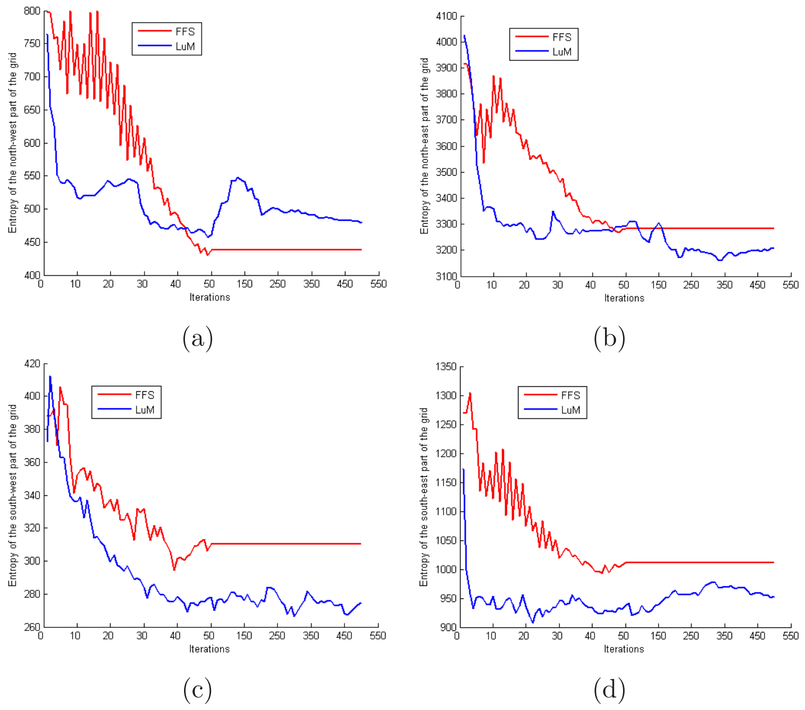
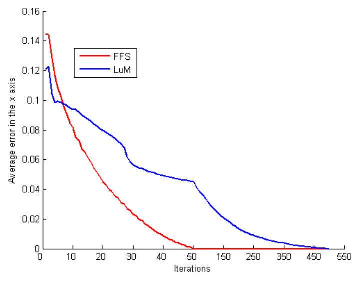


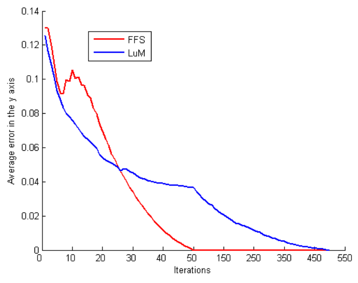
Figure 10.5: (a): $H(m)$ for North-West (top-left quadrant) region of m. (b): $H(m)$ for North-East (top-right quadrant). (c): For South-West (bottom-left). (d): For South-East (bottom-right)

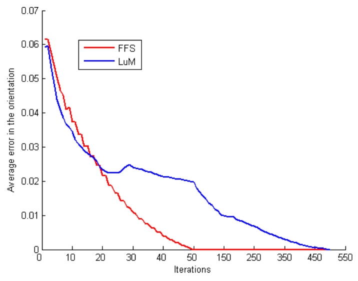
10.5.4 Global Pose Based Estimation

Pose based estimation needs a ground truth reference pose, see section 10.4.2. Since a ground truth for the NIST data set is not available, we just use the final set of poses of each algorithm. This necessarily leads to a graph that converges to an error of zero. Hence it does not give any information about the actual mapping quality, but it shows the behavior of the algorithms in terms of rate of convergence. Fig. 10.6 shows the behavior of the algorithms using error-metrics presented in section 10.4.2.

With respect to path to convergence, the pose based evaluation also shows the same properties of LuM and FFS as the grid based: LuM is ”more monotonic”, while FFS has jittering behavior. Interestingly the pose based evaluation shows FFS converging faster, which is in contrast to the result using grid based evaluation. While reasons for this different result will be topic of future discussion, it again shows that the choice of evaluation method has an influence on the property description of the algorithms.







(a) (b) (c)

Figure 10.6: (a): $E(x)$ for FFS and LuM. (b): $E(y)$. (c): $E(\theta)$ for FFS and LuM. The errors $E(x)$ and $E(y)$ are given in meters, $E(\theta)$ is given in radians.

10.6 Conclusions and Future Work

This paper has presented a performance evaluation of two simultaneous localization and mapping (SLAM) algorithms namely $6D$ Lu/Milios SLAM ($6D$ LUM) and Force Field Simulation (FFS). These two algorithms have been applied to a $2D$ data set, provides by NIST. The results have been compared using two different metrics, i.e., an occupancy grid based method and a pose based method. In addition these metrics have checked by visual inspection for plausibility. $6D$ LUM and FFS show similar performances on the data set considered in this paper.

Needless to say a lot of work remains to be done. The two algorithms have been on one data set. However, in robotic exploration task the environment is the greatest element of uncertainty. Mapping algorithms might fail in certain environments. In future work we plan to benchmark mapping algorithms using more suitable standardized tests and evaluate on automatically generated test cases. The grid and pose based evaluation methods will be used for these evaluations.

Acknowledgement

We thank National Institute of Standards and Technology (NIST), USA for providing us with the Rescue and Maze data sets.

REFERENCES

[1] Scott Joel Aaronson. *Limits on Efficient Computation in the Physical World*. PhD thesis, University of California, Berkeley, Berkeley, CA, USA, 2004.

[2] Y. Adato, Y. Vasilyev, O. Ben Shahar, and T. Zickler. Toward a theory of shape from specular flow. In *ICCV07*, pages 1–8, 2007.

[3] Nagesh Adluru, Longin J. Latecki, Rolf Lakamper, Thomas Young, Xiang Bai, and Ari Gross. Contour grouping based on local symmetry. In *ICCV '07: Proceedings of the Eleventh IEEE International Conference on Computer Vision*. IEEE Computer Society, 2007.

[4] Nagesh Adluru, Longin Jan Latecki, Rolf Lakämper, and Raj Madhavan. Robot mapping for rescue robots. In *Proc. of the IEEE Int. Workshop on Safety, Security and Rescue Robotics (SSRR)*, Gaithersburg, Maryland, USA, August 2006.

[5] Nagesh Adluru, Longin Jan Latecki, Marc Sobel, and Rolf Lakaemper. Merging maps of multiple robots. In *IAPR International Conference on Pattern Recognition (ICPR)*, December 2008.

[6] Nancy M. Amato, Ken A. Dill, and Guang Song. Using motion planning to map protein folding landscapes and analyze folding kinetics of known native structures. In *RECOMB '02: Proceedings of the sixth annual international conference on Computational biology*, pages 2–11, New York, NY, USA, 2002. ACM Press.

[7] Francesco Amigoni, Simone Gasparini, and Maria Gini. Building segment-based maps without pose information. In *Proceedings of the IEEE*, volume 94, pages 1340–1359, July 2006.

[8] C. Andrieu, J. de Freitas, and A. Doucet. Sequential bayesian estimation and model selection applied to neural networks. Technical Report CUED/F-INFENG/TR 341, Cambridge University, 1999.

[9] C. Andrieu, J. de Freitas, and A. Doucet. Sequential mcmc for bayesian model selection, 1999.

[10] M. Ankerst, G. Kastenmüller, H.-P. Kriegel, and T. Seidl. 3D shape histograms for similarity search and classification in spatial databases. In R. Güting, D. Papadias, and F. Lochovsky, editors, *Advances in Spatial Databases, 6th Int. Symposium, SSD'99*, volume 1651, pages 207–228, Hong Kong, China, 1999. Springer.

[11] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Trans. Pattern Anal. Mach. Intell.*, 9(5):698–700, 1987.

[12] C. Aslan and S. Tari. An axis based representation for recognition. In *ICCV*, pages 1339–1346, 2005.

[13] Venkat R. Ayyagari, Faysal Boughorbel, Andreas Koschan, and Mongi A. Abidi. A new method for automatic 3d face registration. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, page 119, Washington, DC, USA, 2005. IEEE Computer Society.

[14] Xiang Bai and Longin Jan Latecki. Path similarity skeleton graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(7), July 2008.

[15] T. Bailey. Mobile robot localisation and mapping in extensive outdoor environments, 2001.

[16] Tim Bailey, Juan Nieto, Jose Guivant, Michael Stevens, and Eduardo Nebot. Consistency of the EKF-SLAM Algorithm. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '06)*, Bejing, China, 2006.

[17] Tim Bailey, Juan Nieto, and Eduardo Nebot. Consistency of the FastSLAM Algorithm. In *IEEE International Conference on Robotics and Automation (ICRA '06))*, Orlando, Florida, U.S.A., 2006.

[18] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24:705–522, 2002.

[19] Sven Bertel, Thomas Barkowsky, Dominik Engel, and Christian Freksa. Computational modeling of reasoning with mental images: Basic requirements. In *Proceedings of the 7th International Conference on Cognitive Modeling*, pages 50–55, 2006.

[20] P.J. Besl and N.D. McKay. A method for registration of $3-D$ shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[21] Peter Biber and Wolfgang Strasser. The normal distributions transform: A new approach to laser scan matching. In *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.

[22] Peter Biber and Wolfgang Strasser. nScan-matching: simultaneous matching of multiple scans and application to slam. In *Robotics and Automation. ICRA Proceedings IEEE International Conference on*, pages 2270– 2276, May 2006.

[23] A. Birk and S. Carpin. Merging occupancy grid maps from multiple robots. In *Proceedings of the IEEE*, volume 94, pages 1384 – 1397, July 2006.

[24] Andreas Birk. Learning geometric concepts with an evolutionary algorithm. In *Proc. of The Fifth Annual Conference on Evolutionary Programming*. The MIT Press, Cambridge, 1996.

[25] M. J. Black and D. J. Fleet. Probabilistic detection and tracking of motion boundaries. *Int. J. of Computer Vision*, 38(3):231–245, 2000.

[26] A. Blake and M. Isard. *Active Contours*. Springer-Verlag, 1997.

[27] H. Blum. A transformation for extracting new descriptors of shape. In W. Wathen-Dunn, editor, *Models for the Perception of Speech and Visual Form*. MIT Press, Cambridge, 1967.

[28] H. Blum. Biological shape and visual science. *Journal of Theor. Biol.*, 38:205–287, 1973.

[29] Eran Borenstein and Shimon Ullman. Class-specific, top-down segmentation. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part II*, pages 109–124, London, UK, 2002. Springer-Verlag.

[30] Gunilla Borgefors. Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(6):849–865, 1988.

[31] D. Borrmann, J. Elseberg, K. Lingemann, A. Nüchter, and J. Hertzberg. The Extension of Lu and Milios Style SLAM to 6 Degree of Freedom. In *IROS 2007, (submitted)*, 2007.

[32] D. Borrmann, J. Elseberg, K. Lingemann, A. Nüchter, and J. Hertzberg. Globally Consistent 3D Mapping with Scan Matching. *Journal of Robotics and Autonomous Sytems*, 2007, (To appear).

[33] F. Boughorbel, A. Koschan, B. Abidi, and M. Abidi. Gaussian fields: a new criterion for 3d rigid registration. *Pattern Recognition*, 37:1567–1571, July 2004.

[34] A. M. Bronstein, M. M. Bronstein, A. M. Bruchstein, and R. Kimmel. Matching two-dimentional articulated shapes using generalized multidimentional scaling. In *Proc. Conf. on Articulated Motion and Deformable Objects*, pages 48–57, 2006.

[35] A. M. Bronstein, M. M. Bronstein, and R. Kimmel. Rock, paper, and scissors: extrinsic vs. intrinsic similarity of non-rigid shapes. In *ICCV*, 2007.

[36] A. M. Bronstein, M. M. Bronstein, and R. Kimmel. Analysis of two-dimentaional non-rigid shapes. *IJCV*, 78:67–88, 2008.

[37] J.W. Bruce, P.J. Giblin, and C. Gibson. Symmetry sets. *Proc. Roy. Soc. Edinburgh*, 101(A):163–186, 1985.

[38] H. Bunke. On a relation between graph edit distance and maximum common subgraph. *Pattern Recogn. Lett.*, 18(9):689–694, 1997.

[39] Horst Bunke, Pasquale Foggia, C. Guidobaldi, Carlo Sansone, and Mario Vento. A comparison of algorithms for maximum common subgraph on randomly connected graphs. In *Proceedings of the Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pages 123–132, London, UK, 2002. Springer-Verlag.

[40] W. Burgard, M. Moors, C. Stachniss, and F. Schneider. Coordinated multi-robot exploration.

[41] Wolfram Burgard, Mark Moors, Dieter Fox, Reid Simmons, and Sebastian Thrun. Collaborative multi-robot exploration. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2000.

[42] James Carpenter, Peter Clifford, and Paul Fearnhead. Building robust simulation-based filters for evolving data sets. Technical report, Dept. of Statistics, University of Oxford, 1999.

[43] S. Carpin, A. Birk, and V. Jucikas. On map merging. *Robotics and Autonomous Systems*, 53(1):1–14, 2005.

[44] S. Carpin and G. Pillonetto. Robot motion planning using adaptive random walks. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, pages 1050–4729, September 2003.

[45] S. Carpin and G. Pillonetto. Motion planning using adaptive random walks. In *Robotics, IEEE Transactions on [see also Robotics and Automation, IEEE Transactions on]*, volume 21, pages 129–136, Feb 2005.

[46] Defense Advanced Research Projects Agency (DARPA) Grand Challenge. `http://www.darpa.mil/grandchallenge/index.asp`, 2007.

[47] H. J. Chang, C. S. G. Lee, Y.-H. Lu, and Y. C. Hu. P-SLAM: Simultaneous localization and mapping with environmental-structure prediction. *IEEE Trans. on Robotics*, 23(2):281–293, 2007.

[48] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image Vision Comput.*, 10(3):145–155, 1992.

[49] Yuanhao Chen, Long Zhu, Chenxi Lin, Alan Yuille, and Hongjiang Zhang. Rapid inference on a novel and/or graph for object detection, segmentation and parsing. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 289–296. MIT Press, Cambridge, MA, 2008.

[50] H. I. Choi, S. W. Choi, and H. P. Moon. Mathematical theory of medial axis transform. *Pacific Journal of Mathematics*, 181(1):57–88, 1997.

[51] Jonathan D. Cohen, Ming C. Lin, Dinesh Manocha, and Madhav K. Ponamgi. I-COLLIDE: An interactive and exact collision detection system for large-scale environments. In *Symposium on Interactive 3D Graphics*, pages 189–196, 218, 1995.

[52] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. Introduction to algorithms. In *MIT Press*, page 2^{nd} edition, 2001.

[53] Timothee Cour, Florence Benezit, and Jianbo Shi. Spectral segmentation with multiscale graph decomposition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1124–1131, Washington, DC, USA, 2005. IEEE Computer Society.

[54] I. Cox, Jim Rehg, and S. Hingorani. A bayesian multiple-hypothesis approach to edge grouping and contour segmentation. *International Journal of Computer Vision*, 11(1):5 – 24, August 1993.

[55] Wolfram Burgard Cyrill Stachniss, Grisetti Giorgio and Nicholas Roy. Analyzing gaussian proposal distributions for mapping with rao-blackwellized particle filters. In *In Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (To appear)*, 2007.

[56] Charles Darwin. On the origin of species. November 1859.

[57] G. Dedeoglu and G. Sukhatme. Landmark-based matching algorithm for cooperative mapping by autonomous robots, 2000.

[58] G. Dissanayake, H. Durrant-Whyte, and T. Bailey. A computationally efficient solution to the simultaneous localization and map building (slam) problem. In *In Proceedings of the IEEE Int. Conference on Robotics & Automation (ICRA)*, pages 1009–1014, April 2000.

[59] G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Trans. of Robotics and Automation*, 2001.

[60] M. W. M. G. Dissanayake, P. M. Newman, H. F. Durrant-Whyte, S. Clark, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotic and Automation (TRA)*, 27(3):229–241, 2001.

[61] R. Douc and O. Cappé. Comparison of resampling schemes for particle filtering. In *Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on*, pages 64–69, Sep 2005.

[62] Arnaud Doucet. *Monte Carlo Methods for Bayesian Estimation of Hidden Markov Models. Application to Radiation Signals.* PhD thesis, Univ. Paris-Sud, Orsay, 1997. in French with chapters 4 and 5 in English.

[63] Arnaud Doucet. On sequential simulation-based methods for bayesian filtering. Technical report, Cambridge University Department of Engineering, 1998.

[64] Arnaud Doucet, Nando de Freitas, and Neil Gordon. *Sequential Monte Carlo Methods in Practice.* Springer-Verlag, 2001.

[65] Arnaud Doucet, Nando de Freitas, Kevin P. Murphy, and Stuart J. Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 176–183, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[66] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes. A taxonomy for multi-agent robotics, 1996.

[67] David W. Eggert, Andrew W. Fitzgibbon, and Robert B. Fisher. Simultaneous registration of multiple range views for use in reverse engineering of CAD models. *Computer Vision and Image Understanding: CVIU*, 69(3):253–272, 1998.

[68] Ahmed Elgammal, Ramani Duraiswami, and Larry S. Davis. Efficient kernel density estimation using the fast gauss transform with applications to color modeling and tracking. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(11):1499–1504, 2003.

[69] A. Eliazar and R. Parr. DP-SLAM: Fast, robust simulataneous localization and mapping without predetermined landmarks. In *Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 2003.

[70] A. Eliazar and R. Parr. DP-SLAM 2.0. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2004.

[71] Austin Eliazar and Roland Parr. Dp-slam 2.0. In *Proc. of IEEE International Conference on Robotics and Automation*, 2004.

[72] Lynne E.Parker. Current state of the art in distributed autonomous mobile robotics. In Lynne E. Parker, George Bekey, and Jacob Barhen, editors, *Distributed Autonomous Robotic System 4*, pages 3–12. Springer-Verlag, Tokyo, October 2000. ISBN: 4-431-70295-4.

[73] W. Eric and L. Grimson. *Object recognition by computer: the role of geometric constraints.* MIT Press, Cambridge, MA, USA, 1990.

[74] Sebastian Thrun et. al. Stanley: The robot that won the darpa grand challenge: Research articles. *J. Robot. Syst.*, 23(9):661–692, 2006.

[75] The RoboCup Federation. `http://www.robocup.org/`, 2007.

[76] Jacob Feldman. Does vision work? towards a semantics of perception. In E. Lepore and Z. Pylyshyn, editors, *What is Cognitive Science?*, pages 208–229. Basil Blackwell, 1999.

[77] Jacob Feldman. The role of objects in perceptual grouping. *Acta Psychologica*, 102(2-3):137–163, September 1999.

[78] Jacob Feldman. Minimization of boolean complexity in human concept learning. *Nature*, 407:630–633, 2000.

[79] Jacob Feldman. Bayesian contour integration. *Perception & Psychophysics*, 63(7):1171–1182, 2001.

[80] Jacob Feldman. Perceptual grouping by selection of a logically minimal model. *Int. J. Comput. Vision*, 55(1):5–25, 2003.

[81] Jacob Feldman. Formation of visual objects in the early computation of spatial relations. *Perception & Psychophysics*, 69(5):816–827, 2007.

[82] Jacob Feldman and Manish Singh. Bayesian estimation of the shape skeleton. In *Proc. of the National Academy of Sciences*, volume 103 (47), pages 18014–18019, November 2006.

[83] Pedro Felzenszwalb and David McAllester. A min-cover approach for finding salient curves. In *CVPRW: Proceedings of the Conference on Computer Vision and Pattern Recognition Workshop*, Washington, DC, USA, 2006. IEEE Computer Society.

[84] Pedro F. Felzenszwalb and Joshua Schwartz. Hierarchical matching of deformable shapes. In *CVPR*, 2007.

[85] J.W. Fenwick, P.M. Newman, and J.J. Leonard. Cooperative concurrent mapping and localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 1810–1817, August 2002.

[86] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(1):36–51, 2008.

[87] V. Ferrari, F. Jurie, and C. Schmid. Accurate object detection with deformable shape models learnt from images. In *CVPR*, pages 1–8, 2007.

[88] Vittorio Ferrari, Tinne Tuytelaars, and Luc. J. Van. Gool. Object detection by contour segment networks. In *ECCV*, pages 14–28, 2006.

[89] FGAN. `http://www.elrob2006.org/`, 2007.

[90] D. J. Field, A. Hayes, and R.F. Hess. Contour integration by the human visual system: evidence for a local association field. *Vision Research*, 33:173–193, January 1993.

[91] A. Fitzgibbon. Robust registration of 2d and 3d point sets. In *Proc. British Machine Vision Conference, volume II, Manchester, UK*, pages 411–420, 2001.

[92] Robert Floyd and Richard Beigel. *The language of machines: an introduction to computability and formal languages*. Computer Science Press, Inc., New York, NY, 1994.

[93] D. Fox, W. Burgard, H. Kruppa, and S. Thrun. Efficient multi-robot localization based on monte carlo approximation, 1999.

[94] D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, and B. Stewart. Distributed multirobot exploration and mapping. In *Proceedings of the IEEE*, volume 94, pages 1325–1339, July 2006.

[95] Dieter Fox. Adapting the sample size in particle filters through kld-sampling. *International Journal of Robotics Research (IJRR)*, 22(12):985–1003, 2003.

[96] C. Freksa, M. Knauff, B. Krieg-Brckner, B. Nebel, and Th. Barkowsky. *Spatial Cognition IV, Reasoning, Action, Interaction*. Springer, October 2004.

[97] Udo Frese. Treemap: An o(log n) algorithm for indoor simultaneous localization and mapping. *Auton. Robots*, 21(2):103–122, 2006.

[98] Udo Frese, Per Larsson, and Tom Duckett. A multilevel relaxation algorithm for simultaneous localization and mapping. *Robotics, IEEE Transactions on Robotics and Automation*, 21:196–207, April 2005.

[99] S. A. Friedberg. Finding axes of skewed symmetry. *CVGIP*, 34:138–155, 1986.

[100] Meirav Galun, Ronen Basri, and Achi Brandt. Multiscale edge detection and fiber enhancement using differences of oriented means. In *ICCV '07: Proceedings of the Eleventh IEEE International Conference on Computer Vision*. IEEE Computer Society, 2007.

[101] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[102] D. Geiger, T. Liu, and R. V. Kohn. Representation and self-similarity of shapes. *IEEE Trans. PAMI*, 25:86–100, 2003.

[103] Andrew Gelman. Iterative and non-iterative simulation algorithms. In *Computing Science and Statistics: Proceedings of the 24^{th} Symposium on the Interface*, pages 433–438, 1992.

[104] Thea Ghiselli-Crippa, Stephen C. Hirtle11, and Paul Munro. *Connectionist Models in Spatial Cognition*, volume 32, pages 87–104. Springer, Netherlands, 1996.

[105] Peter J. Giblin. Symmetry sets and medial axes in two and three dimensions. In *Proceedings of the 9th IMA Conference on the Mathematics of Surfaces*, pages 306–321, London, UK, 2000. Springer-Verlag.

[106] Peter J. Giblin and Benjamin B. Kimia. On the intrinsic reconstruction of shape from its symmetries. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(7):895–911, 2003.

[107] Peter J. Giblin and Benjamin B. Kimia. On the local form and transitions of symmetry sets, medial axes, and shocks. *Int. J. Comput. Vision*, 54(1-3):143–156, 2003.

[108] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *Radar and Signal Processing, IEE Proceedings of*, volume 140, pages 107–113, April 1993.

[109] Kristen Grauman and Trevor Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision*, pages 1458–1465, Washington, DC, USA, 2005. IEEE Computer Society.

[110] P. Green. Reversible jump markov chain monte carlo computation and bayesian model determination, 1995.

[111] L. Greengard and V. Rokhlin. A Fast Algorithm for Particle Simulations. *Journal of Computational Physics*, 73:325–348, 1987.

[112] Leslie Greengard and John Strain. The fast gauss transform. *SIAM J. Sci. Stat. Comput.*, 12(1):79–94, 1991.

[113] W. E. L. Grimson. The combinatorics of object recognition in cluttered environments using constrained search. *Artif. Intell.*, 44(1-2):121–165, 1990.

[114] W. E. L. Grimson and T. Lozano-Pérez. Localizing overlapping parts by searching the interpretation tree. *IEEE Trans. Pattern Anal. Mach. Intell.*, 9(4):469–482, 1987.

[115] W. Eric L. Grimson. *Object recognition by computer: the role of geometric constraints*. MIT Press, Cambridge, MA, USA, 1990.

[116] G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based slam with raoblackwellized particle filters by adaptive proposals and selective resampling, 2005.

[117] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with raoblackwellized particle filters. *IEEE Trans. on Robotics*, 23:34–46, 2007.

[118] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *In Proceedings of the IEEE Int. Conference on Robotics & Automation (ICRA)*, pages 2443–2448, 2005.

[119] T. Hagadone. Molecular substructure similarity searching: efficient retrieval in two-dimensional structure databases. *J. Chem. Inf. Comput. Sci.*, 32:515–521, 1992.

[120] D. Hähnel, D. Schulz, and W. Burgard. Map building with mobile robots in populated environments. In *Proceedings of International Conference on Intelligent Robots and Systems (IROS'02)*, 2002.

[121] Feng Han and Song-Chun Zhu. Bottom-up/top-down image parsing by attribute graph grammar. In *Proceedings of the Tenth IEEE International Conference on Computer Vision*, pages 1778–1785, Washington, DC, USA, 2005. IEEE Computer Society.

[122] J. E. Handschin. Monte carlo techniques for prediction and filtering of non-linear stochastic processes. *Automatica*, 6:555–563, 1970.

[123] J. E. Handschin and D. Q. Mayne. Monte carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering. *Internation Journal of Controls*, 9(5):353–358, 1969.

[124] Derek Hoiem, Andrew Stein, Alexei A. Efros, and Martial Hebert. Recovering occlusion boundaries from a single image. In *International Conference on Computer Vision (ICCV)*, October 2007.

[125] Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, 4(4):629–642, 1987.

[126] Berthold K. P. Horn, Hugh M. Hilden, and Shahriar Negahdaripourt. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America*, 5(7):1127–1135, 1988.

[127] P. V. C. Hough. Methods and means for recognizing complex patterns, 1962.

[128] A. Howard and N. Roy. Radish: The Robotics Data Set Repository, Standard data sets for the robotics community. `http://radish.sourceforge.net/`, 2003 – 2006.

[129] Andrew Howard. Multi-robot simultaneous localization and mapping using particle filters. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[130] Shoudong Huang and G. Dissanayake. Convergence analysis for extended kalman filter based slam. In *In Proceedings of the IEEE Int. Conference on Robotics & Automation (ICRA)*, pages 412–417, May 2006.

[131] Wesley H. Huang and Kristopher R. Beevers. Topological map merging. *Int. J. Rob. Res.*, 24(8):601–613, 2005.

[132] Luca Iocchi, Daniele Nardi, and Massimiliano Salerno. Reactivity and deliberation: A survey on multi-robot systems. In *Balancing Reactivity and Social Deliberation in Multi-Agent Systems, From RoboCup to Real-World Applications (selected papers from the ECAI 2000 Workshop and additional contributions)*, pages 9–34, London, UK, 2001. Springer-Verlag.

[133] Michael Isard and Andrew Blake. Condensation – conditional density propagation for visual tracking. *Int. Journal of Computer Vision*, 29(1):5–28, 1998.

[134] A. Jacoff, E. Messina, and J. Evans. Performance evaluation of autonomous mobile robots. *Industrial Robot: An Int. Journal*, 29(3), 2002.

[135] Adam Jacoff, E. Messina, and B. Weiss. Evolution of a performance metric for urban search and rescue robots. In *Proc. of the Performance Metrics for Intelligent Systems (PerMIS) Workshop*, Gaithersburg, MD, 2003.

[136] Andrei C. Jalba, Michael H.F. Wilkinson, and Jos B.T.M. Roerdink. Cpm: A deformable model for shape recovery and segmentation based on charged particles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1320–1335, 2004.

[137] S. Julier and J. Uhlmann. A general method for approximating nonlinear transformations of probability distributions, 1996.

[138] S.J. Julier. The scaled unscented transformation. In *American Control Conference, 2002. Proceedings of the 2002*, volume 6, pages 4555–4559, November 2002.

[139] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[140] T. Kanade. Recovery of the three-dimensional shape of an object from a single view. *Artificial Intelligence*, 17:409–460, 1981.

[141] L.E. Kavraki, P. Svestka, J.-C. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensionalconfiguration spaces. In *Robotics and Automation, IEEE Transactions on*, volume 12, pages 566–580, Aug 1996.

[142] Benjamin B. Kimia and Amir Tamrakar. The role of propagation and medial geometry in human vision. In *BMCV '02: Proceedings of the Second International Workshop on Biologically Motivated Computer Vision*, pages 219–229, London, UK, 2002. Springer-Verlag.

[143] David C. Knill and Whitman Richards, editors. *Perception as Bayesian inference.* Cambridge University Press, New York, NY, USA, 1996.

[144] J. Ko, B. Stewart, D. Fox, K. Konolige, and B. Limketkai. A practical, decision-theoretic approach to multi-robot mapping and exploration, 2003.

[145] A. Kong, J.S. Liu, and W.H. Wong. Sequential imputations and bayesian missing data problems. *Amer. Stat. Assoc.*, 89:278–288, 1994.

[146] K. Konolige, D. Fox, B. Limketkai, J. Ko, , and B. Stewart. Map merging for distributed robot navigation. In *Intl. Conf. on Intelligent Robots and Systems*, pages 212–217, 2003.

[147] Kurt Konolige. Map merging for distributed robot navigation. In *Proceedings of the 2003 IEEE International Conference on Intelligent Robots and Systems*, pages 212–217, Las Vegas, NV, October 2003.

[148] Kurt Konolige and Ken Chou. Markov localization using correlation. In *IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1154–1159, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[149] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. School of Computer Science & Software Engineering, The University of Western Australia, 2008. Available from: <http://www.csse.uwa.edu.au/∼pk/research/matlabfns/>.

[150] G. J. M. Kruijff, H. Zender, P. Jensfelt, and H. I. Christensen. Situated dialogue and spatial organization: What, where and why? *Int. J. of Advanced Robotic Systems*, 4(1):125–138, 2007.

[151] A. Kuijper and O.F. Olsen. Describing and matching 2d shapes by their points of mutual symmetry. In *ECCV06*, pages III: 213–225, 2006.

[152] Arjan Kuijper and Ole Fogh Olsen. Transitions of the pre-symmetry set. In *ICPR '04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 3*, pages 190–193, Washington, DC, USA, 2004. IEEE Computer Society.

[153] Arjan Kuijper and Ole Fogh Olsen. Geometric skeletonization using the symmetry set. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, pages 497–500, September 2005.

[154] Arjan Kuijper, Ole Fogh Olsen, Philip Bille, and Peter Giblin. Matching 2d shapes using their symmetry sets. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*, pages 179–182, Washington, DC, USA, 2006. IEEE Computer Society.

[155] Arjan Kuijper, Ole Fogh Olsen, Peter Giblin, and Mads Nielsen. Alternative 2d shape representations using the symmetry set. *J. Math. Imaging Vis.*, 26(1-2):127–147, 2006.

[156] B. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119:191–233, 2000.

[157] B. J. Kuipers. The Spatial Semantic Hierarchy. *Artificial Intelligence*, 119:191–233, 2000. http://www.cs.utexas.edu/users/qr/papers/Kuipers-aij-00.html.

[158] Benjamin J. Kuipers. The cognitive map: Could it have been any other way? In H. L. Pick and L. P. Acredolo, editors, *Spatial Orientation: Theory, Research, and Application*, pages 345–360. Plenum Press, New York, 1983.

[159] M. P. Kumar, P. H. S. Torr, and A. Zisserman. OBJ CUT. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, volume 1, pages 18–25, 2005.

[160] Dmitry Lagunovsky and Sergey Ablameyko. Fast line and rectangle detection by clustering and grouping. In *CAIP '97: Proceedings of the 7th International Conference on Computer Analysis of Images and Patterns*, pages 503–510, London, UK, 1997. Springer-Verlag.

[161] R. Lakaemper, N. Adluru, L. J. Latecki, and R. Madhavan. Multi Robot Mapping using Force Field Simulation. *Journal of Field Robotics, Special Issue on Quantitative Performance Evaluation of Robotic and Intelligent Systems. (To appear)*, 2007.

[162] R. Lakaemper and L. J. Latecki. Decomposition of 3d laser range data using planar patches. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2006.

[163] R. Lakaemper, L. J. Latecki, and D. Wolter. Incremental multi-robot mapping. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2005.

[164] R. Lakaemper, A. Nüchter, N. Adluru, and L. J. Latecki. Performance of 6d LuM and FFS SLAM: An example for comparison using grid and pose based evaluation methods. In *Workshop on Performance Metrics and Intelligent Systems (PerMIS)*, Gaithersburg, MD, August 2007.

[165] Rolf Lakaemper, Nagesh Adluru, and Longin Jan Latecki. Force field based n-scan alignment. In *European Conference on Mobile Robots*, Freiburg, Germany, September 2007.

[166] Rolf Lakaemper, Nagesh Adluru, Longin Jan Latecki, and Raj Madhavan. Multi robot mapping using force field simulation: Research articles. *J. Field Robot.*, 24(8-9):747–762, 2007.

[167] Rolf Lakaemper, Longin Jan Latecki, Xinyu Sun, and Diedrich Wolter. *Geometric Robot Mapping*, volume 3429, pages 11–22. Springer, Berlin, Heidelberg, Germany, 2005.

[168] L. J. Latecki and R. Lakaemper. Polygonal approximation of laser range data based on perceptual grouping and em. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2006.

[169] L. J. Latecki and R. Lakamper. Convexity rule for shape decomposition based on discrete contour evolution. *CVIU*, 73:441–454, 1999.

[170] L. J. Latecki, M. Sobel, and R. Lakaemper. New EM derived from Kullback-Leibler divergence. In *ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 2006.

[171] Longin Jan Latecki, ChengEn Lu, Marc Sobel, and Xiang Bai. Multiscale random fields with application to contour grouping. In *Neural Information Processing Systems Conf. (NIPS), Vancouver*, December 2008.

[172] J. Latombe. Motion planning: A journey of robots, molecules, digital actors, and other artifacts, 1999.

[173] B. Leibe and B. Schiele. Interleaved object categorization and segmentation. In *BMVC03*, 2003.

[174] Bastian Leibe, Edgar Seemann, and Bernt Schiele. Pedestrian detection in crowded scenes. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 878–885, Washington, DC, USA, 2005. IEEE Computer Society.

[175] M. Leyton. *Symmetry, Causality, Mind.* MIT Press, Cambridge, 1992.

[176] H. Ling and D. W. Jacobs. Shape classification using inner-distance. *IEEE Trans. PAMI*, 29:286–299, 2007.

[177] J. S. Liu. Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statist. Comput.*, 6:113–119, 1996.

[178] J. S. Liu, R. Chen, and W. H. Wong. Rejection control and sequential importance sampling. *Journal of the American Statistical Association*, 93(443):1022–1031, 1998.

[179] T. Liu, D. Geiger, and A. L. Yuille. Segmenting by seeking the symmetry axis. In *Proc. CVPR*, pages 994–998, 1998.

[180] E. H. Lockwood. *A Book of Curves.* Cambridge University Press, 2007.

[181] D. G. Lowe. *Perceptual Organization and Visual Recognition.* Kluwer Academic Publishers, Boston, 1985.

[182] David G. Lowe. Object recognition from local scale-invariant features. In *ICCV99*, pages 1150–1157, 1999.

[183] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Auton. Robots*, 4(4):333–349, 1997.

[184] F. Lu and E. Milios. Globally Consistent Range Scan Alignment for Environment Mapping. *Autonomous Robots*, 4(4):333 – 349, October 1997.

[185] F. Lu and E. Milios. Robot pose estimation in unknown environments by matching 2D range scans. *Journal of Intelligent and Robotic Systems*, 1997.

[186] Tanner M.A. and Wong W.H. The calculation of posterior distirbutions by data augmentation. *Amer. Stat. Assoc.*, 82:528–550, 1987.

[187] S. MacEachern, M. Clyde, and J. Liu. Sequential importance sampling for nonparametric bayes models: The next generation, 1998.

[188] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[189] A. Martinelli, A. Tapus, K.O. Arras, and R. Siegwart. Multi-resolution slam for real world navigation. In *Proceedings of the 11th International Symposium of Robotics Research*, 2003.

[190] Graham McNeill and Sethu Vijayakumar. Part-based probabilistic point matching using equivalence constraints. In *Neural Information Processing Systems*, pages 969–976, 2006.

[191] Marina Meilă. Comparing clusterings: an axiomatic view. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 577–584, New York, NY, USA, 2005. ACM.

[192] E. Menegatti and E. Pagello. Omnidirectional distributed vision for multi-robot mapping, June 2002. In Proc. of the Intern. Symposium on Distributed Autonomous Robotic Systems (DARS02) (TO APPEAR), Fukuoka, Japan.

[193] J. Minguez, L Montesano, and F Lamiraux. Metric-based iterative closest point scan matching for sensor displacement estimation. *Robotics, IEEE Transactions on Robotics*, pages 1047–1054, October 2006.

[194] R. Mohan and R. Nevatia. Perceptual organization for scene segmentation and description. *IEEE Trans. on PAMI*, 14(6):616–635, 1992.

[195] M. Montemerlo, N. Roy, and S. Thrun. Perspectives on standardization in mobile robot programming: The carnegie mellon navigation (CARMEN) toolkit. In *Proceedings of the Conference on Intelligent Robots and Systems (IROS)*, 2003.

[196] M. Montemerlo, N. Roy, S. Thrun, D. Hahnel, C. Stachniss, and J.Glover. Carmen-the carnegie mellon robot navigation toolkit., 2002.

[197] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: a factored solution to simultaneous mapping and localization problem. In *Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 593–598, 2002.

[198] Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Eighteenth national conference on Artificial intelligence*, pages 593–598, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[199] P. Del Moral and L. Miclo. Branching and interacting particle systems approximations of feynman-kac formulae with applications to nonlinear filtering. *Seminaire de Probabilities XXXIV, of Lecture Notes in Mathematics*, 1729:1–145, 2000.

[200] Hans Moravec. Sensor fusion in certainty grids for mobile robots. *AI Mag.*, 9(2):61–74, 1988.

[201] E. W. Myers. An o(ND) difference algorithm and its variations. *Algorithmica*, 1(2):251–266, 1986.

[202] P. Newman, J. Leonard, J.D. Tardos, and J. Neira. Explore and return: experimental validation of real-time concurrent mapping and localization. In *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, pages 1802–1809, August 2002.

[203] M. Novotni and R. Klein. A geometric approach to 3d object comparison. In *Shape Modeling and Applications, SMI 2001 International Conference on.*, pages 167–175, May 2001.

[204] N.Paragios, M.Rousson, and V.Ramesh. Non-rigid registration using distance functions. In *Computer Vision and Image Understanding*, volume 89, pages 142–165, 2003.

[205] A. Nüchter, K. Lingemann, J. Hertzberg, H. Surmann, K. Pervölz, M. Hennig, K. R. Tiruchinapalli, R. Worst, and T. Christaller. Mapping of rescue environments with kurt3d. In *Proc. of the IEEE Int. Workshop on Safety, Security and Rescue Robotics (SSRR)*, Kobe, Japan, June 2005.

[206] Edwin Olson, John Leonard, and Seth Teller. Fast iterative optimization of pose graphs with poor initial estimates. In *Proceedings of ICRA*, pages 2262–2269, 2006.

[207] A. Opelt, A. Pinz, and A. Zisserman. A boundary-fragment-model for object detection. In *Proceedings of the European Conference on Computer Vision*, 2006.

[208] Alex P. Pentland. Perceptual organization and the representation of natural form. *Artif. Intell.*, 28(3):293–331, 1986.

[209] P. Perez, A. Blake, and M. Gangnet. Rjetstream: Probabilistic contour extraction with particles. In *Proc. ICCV*, pages 524–531, 2001.

[210] Michael K. Pitt and Neil Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–630, 1999.

[211] Z. Pizlo, M. Salach-Colyska, and A. Rosenfeld. Curve detection in a noisy image. *Vision Research*, 37:1217–1241, 1997.

[212] Joshua Podolak, Philip Shilane, Aleksey Golovinskiy, Szymon Rusinkiewicz, and Thomas Funkhouser. A planar-reflective symmetry transform for 3D shapes. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 25(3), July 2006.

[213] J. Ponce. On characterizing ribbons and finding skewed symmetries. *CVGIP*, 52:328–340, 1990.

[214] A. P.Witkin and J. M. Tenenbaum. On the role of structure in vision. In J. Beck, B. Hope, and A. Rosenfeld, editors, *Human and Machine Vision*. Academic Press, New York, 1983.

[215] D. Raviv, A. M. Bronstein, M. M. Bronstein, and R. Kimmel. Symmetries of non-rigid shapes. In *Proc. Workshop on Non-rigid Registration and Tracking through Learning*, 2007.

[216] D. Raviv, A.M. Bronstein, M.M. Bronstein, and R. Kimmel. Symmetries of non-rigid shapes. In *ICCV*, October 2007.

[217] Donald B. Reid. An algorithm for tracking multiple targets. *Automatic Control, IEEE Transactions on*, 24(6):843–854, December 1979.

[218] Ioannis M. Rekleitis, Gregory Dudek, and Evangelos E. Milios. Multi-robot collaboration for robust exploration. *Annals of Mathematics and Artificial Intelligence*, 31(1-4):7–40, 2001.

[219] X. Ren, A. Berg, and J. Malik. Recovering human body configurations using pairwise constraints between parts. In *Proc. ICCV*, 2005.

[220] Xiaofeng Ren, Charless C. Fowlkes, and Jitendra Malik. Learning probabilistic models for contour completion in natural images. *Int. J. Comput. Vision*, 77(1-3):47–63, 2008.

[221] C. Robertson and R. Fisher. Parallel evolutionary registration of range data. In *Computer Vision and Image Understanding*, volume 87, pages 39–50, 2002.

[222] S.I. Roumeliotis and I.M. Rekleitis. Analysis of multirobot localization uncertainty propagation. In *Intelligent Robots and Systems, Proceedings. IEEE/RSJ International Conference on*, volume 2, pages 1763–1770, December 2003.

[223] Donald B. Rubin. A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions missing information are modest: the sir algorithm. *Amer. Stat. Assoc.*, 82:543–546, 1987.

[224] Donlad B. Rubin. Using the sir algorithm to simulate posterior distributions. *Bayesian Statistics*, 8:395–402, 1988.

[225] Szymon Rusinkiewicz, Benedict Brown, and Michael Kazhdan. 3D scan matching and registration, *ICCV Short Course*, 2005.

[226] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the ICP algorithm. In *Proceedings of the Third Intl. Conf. on 3D Digital Imaging and Modeling*, pages 145–152, 2001.

[227] Thrun S., Koller D., Ghahramani Z., Durrant-Whyte H., and Ng A.Y. Simultaneous mapping and localization with sparse extended information filters. In J.-D. Boissonnat, J. Burdick, K. Goldberg, and S. Hutchinson, editors, *Proceedings of the Fifth International Workshop on Algorithmic Foundations of Robotics*, Nice, France, 2002. Forthcoming.

[228] Thomas B. Sebastian, Philip N. Klein, and Benjamin B. Kimia. Recognition of shapes by editing their shock graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):550–571, 2004.

[229] A. Shaashua and S. Ullman. Structual saliency: the detection of global salient structures using a locally connected network. In *ICCV*, pages 321–327, 1988.

[230] Jamie Shotton, Andrew Blake, and Roberto Cipolla. Contour-based learning for object detection. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pages 503–510, Washington, DC, USA, 2005. IEEE Computer Society.

[231] K. Siddiqi, B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker. Shapes, shocks and wiggles. *Image and Vision Computing Journal*, 17:365–373, 1999.

[232] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker. Shock graphs and shape matching. *Int. J. Comput. Vision*, 35(1):13–32, 1999.

[233] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker. Shock graphs and shape matching. *Int. J. of Computer Vision*, 35:13–32, 1999.

[234] Kaleem Siddiqi and Benjamin B. Kimia. Parts of visual form: Computational aspects. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(3):239–251, 1995.

[235] Kaleem Siddiqi and Stephen M. Pizer. *Medial Representations: Mathematics, Algorithms and Applications*. Springer-Verlag, 2007.

[236] Reid G. Simmons, David Apfelbaum, Wolfram Burgard, Dieter Fox, Mark Moors, Sebastian Thrun, and Hakan Younes. Coordination for multi-robot exploration and mapping. In *AAAI/IAAI*, pages 852–858, 2000.

[237] A.F.M. Smith and A.E. Gelfand. Bayesian statistics without tears: A sampling-resampling perspective. *Amer. Stat. Assoc.*, 46:84–88, 1992.

[238] G. Song and N. Amato. A motion-planning approach to folding: From paper craft to portein folding. *IEEE Transactions on Robotics and Automation*, 20(1):60–71, 2004.

[239] C. Stachniss, G. Grisetti, and W. Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Proc. of Robotics: Science and Systems (RSS)*, pages 65–72, Cambridge, MA, USA, 2005.

[240] C. Stachniss, D. Hähnel, W. Burgard, and G. Grisetti. On actively closing loops in grid-based FastSLAM. *Advanced Robotics*, 19(10):1059–1080, 2005.

[241] J. S. Stahl and S. Wang. Globally optimal grouping for symmetric boundaries. In *Proc. CVPR*, 2006.

[242] Andrew Stein, Derek Hoiem, and Martial Hebert. Learning to find object boundaries using motion cues. In *IEEE International Conference on Computer Vision (ICCV)*, October 2007.

[243] B. Stewart, J. Ko, D. Fox, and K. Konolige. A hierarchical bayesian approach to mobile robot map structure estimation. In *Conf. on Uncertainty in AI (UAI)*, 2003.

[244] Amir Tamrakar and Benjamin B. Kimia. No grouping left behind: From edges to curve fragments. In *ICCV '07: Proceedings of the Eleventh IEEE International Conference on Computer Vision*. IEEE Computer Society, 2007.

[245] H. Tanizaki. On the nonlinear and nonnormal filter using rejection sampling. *Automatic Control, IEEE Transactions on*, 44:314–319, February 1999.

[246] Hisashi Tanizaki. Nonlinear and nonnormal filters using monte carlo methods. *Comput. Stat. Data Anal.*, 25(4):417–439, 1997.

[247] A. Thayananthan, B. Stenger, P. H. S. Torr, and R. Cipolla. Shape context and chamfer matching in cluttered scenes. In *CVPR*, pages 127–133, 2003.

[248] S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, 20(5):335–363, 2001.

[249] S. Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

[250] S. Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

[251] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press Cambridge, 2005.

[252] S. Thrun, D. Fox, and W. Burgard. A real-time algorithm for mobile robot mapping with application to multi robot and 3D mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '00)*, San Francisco, U.S.A, April 2000.

[253] Nhon Trinh and Benjamin B. Kimia. A symmetry-based generative model for shape. In *ICCV '07: Proceedings of the Eleventh IEEE International Conference on Computer Vision*. IEEE Computer Society, 2007.

[254] Zhuowen Tu, Xiangrong Chen, Alan Yuille, and Song-Chun Zhu. Image parsing: Unifying segmentation, detection, and recognition. *International Journal of Computer Vision*, 63:113–140, July 2005.

[255] Z.W. Tu and A.L. Yuille. Shape matching and recognition:using generative models and informative features. In *ECCV04*, volume 3, pages 195–209, 2004.

[256] Alan Mathison Turing. On computable numbers, with an application to the entscheidungsproblem. In *London Mathematical Society 2(42)*, pages 230–65, 1936.

[257] David H. Uttal. Seeing the big picture: Map use and the development of spatial cognition. *Developmental Science*, 3(3):247–264, August 2000.

[258] Rudolph van der Merwe, Nando de Freitas, Arnaud Doucet, and Eric Wan. The unscented particle filter. Technical Report CUED/F-INFENG/TR380, Cambridge University Engineering Department, August 2000.

[259] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[260] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *IJCV*, 62(1-2):61–81, April 2005.

[261] Shrihari Vasudevan, Viet Nguyen, and R. Siegwart. Towards a cognitive probabilistic representation of space for mobile robots. In *Information Acquisition, IEEE International Conference on*, pages 353–359, August 2006.

[262] Olga Veksler. Star shape prior for graph-cut image segmentation. In *ECCV08*, 2008.

[263] R. Veltkamp and M. Hagedoorn. State-of-the-art in shape matching. Technical Report UU-CS-1999-27, Utrecht University, The Netherlands, 1999.

[264] Hongzhi Wang and John Oliensis. A global contour measure for image segmentation. In *POCV06*, 2006.

[265] Hongzhi Wang and John Oliensis. Shape matching by segmentation expectation. In *ECCV08*, 2008.

[266] L.M. Wang, J.B. Shi, G. Song, and I.F. Shen. Object detection combining recognition and segmentation. In *ACCV07*, pages 189–199, 2007.

[267] Yu Wang and Carsten Maple. A novel efficient algorithm for determining maximum common subgraphs. In *Information Visualisation, 2005. Proceedings. Ninth International Conference on*, pages 657–663, July 2005.

[268] Smith TF Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[269] M. Wertheimer. Untersuchungen zur lehre von der gestalt ii. *Psycologische Forschung*, 4:301–350, 1923.

[270] M. Wertheimer. Principles of perceptual organization. In D.C. Beardslee and M. Wertheimer, editors, *Readings in Perception*, pages 115–135. Princeton: NJ, 1923/1958.

[271] M. Werthimer. Principles of perceptual organization. In *Beardslee, D. C. and Wertheimer, M. (Eds.), Readings in Perception*, pages 115–135, 1923/1958.

[272] J. Willamowski, D. Arregui, G. Csurka, C. Dance, and L. Fan. Categorizing nine visual classes using local appearance descriptors. In *ICPR Workshop Learning for Adaptable Visual Systems*, 2004.

[273] S. Williams. Efficient solutions to autonomous mapping and navigation problems, 2001.

[274] S. Williams, G. Dissanayake, and H. Durrant-Whyte. An efficient approach to the simultaneous localisation and mapping problem, 2002.

[275] S.B. Williams, G. Dissanayake, and H. Durrant-Whyte. Towards multi-vehicle simultaneous localisation and mapping. In *Robotics and Automation, Proceedings. IEEE International Conference on*, volume 3, pages 2743–2748, August 2002.

[276] Richard C. Wilson and Edwin R. Hancock. Structural matching by discrete relaxation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(6):634–648, 1997.

[277] C. Xu and J. Prince. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, pages 359–369, March 1998.

[278] Ronghua Yang, Majid Mirmehdi, and Xianghua Xie. A charged active contour based on electrostatics. In *Advanced Concepts for Intelligent Vision Systems, ACIVS*, pages 173–184. Springer-Verlag LNCS 4179, September 2006.

[279] Xingwei Yang, Xiang Bai, Longin Jan Latecki, and Zhuowen Tu. Improving shape retrieval by learning graph transduction. In *ECCV08*, 2008.

[280] W.K. Yeap1 and M.E. Jefferies. On early cognitive mapping. *Spatial Cognition and Computation*, 2(2):85–116, June 2000.

[281] Stella Yu. Segmentation using multiscale cues. In *CVPR*, pages 247–254, 2004.

[282] H. Zabrodsky, S. Peleg, and D. Avnir. A measure of symmetry based on shape similarity. In *CVPR*, pages 703–706, 1992.

[283] V.S. Zaritskii, V.B. Svetnik, and L.I. Shimelevich. Monte carlo technique in problems of optimal data processing. *Auto. Remo. Cont.*, 12:95–103, 1975.

[284] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *Int. J. Comput. Vision*, 73(2):213–238, 2007.

[285] Qihui Zhu, Gang Song, and Jianbo Shi. Untangling cycles for contour grouping. In *ICCV '07: Proceedings of the Eleventh IEEE International Conference on Computer Vision*. IEEE Computer Society, 2007.

[286] Qihui Zhu, Liming Wang, Yang Wu, and Jianbo Shi. Contour context selection for object detection: A set-to-set contour matching approach. In *ECCV08*, 2008.

[287] S. C. Zhu and A. Yuille. Forms: a flexible object recognition and modelling system. In *Proc. ICCV*, pages 465–472, 1995.

[288] Song-Chun Zhu, , Yun Nan Wu, and David Mumford. Filters, random-fields and maximum-entropy (frame): Towards a unified theory for texture modeling. *IJCV*, 27(2):107–126, March 1998.

[289] Song-Chun Zhu, Cheng-En Guo, Yizhou Wang, and Zijian Xu. What are textons? *Int. J. Comput. Vision*, 62(1-2):121–143, 2005.

[290] Song-Chun Zhu and David Mumford. *A Stochastic Grammar oF Images.* Now Publishers, 2007.

[291] Todd Zickler, Ravi, Ramamoorthy, Sebastian Enrique, and Peter N. Belhumeur. Reflectance sharing : Predicting appearance from a sparse set of images of a known shape. *PAMI*, 28(8):1287–1302, 2006.

[292] R. Zlot, A. Stentz, M. Dias, and S. Thayer. Multi-robot exploration controlled by a market economy. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3016–3023, May 2002.

APPENDIX A

Some Basic Particle Filter Derivations in Robotics

A.1 Sequential Monte Carlo estimations or in other words PARTICLE FILTERS

BASIC IDEA: Instead of heuristically assuming features about distributions and tracking them simulate distributions using randomly drawn samples (**particles**) from the distributions.

The drawn particles can be used to for a functional estimate ($E(f(x))$) as:

$$E(f(x)) = \int_x f(x)p(x|y)dx \tag{A.1}$$

$$\approx \frac{1}{N}\sum_{i=1}^{N} f(x^{(i)}) \text{ where } x \text{ i.i.d from } p(x|y) \tag{A.2}$$

As $N \to \infty$ the simulation converges to true estimate.

ADVANTAGE: Gives power to filter non-linear, non-Gaussian without working out complicated analytical math governing the processes and gives us computational feasibility.

BIGGEST HURDLE: Drawing samples from a distribution and trying to estimate it is a chicken-egg problem! So we constantly *have* to iterate between updating the distribution and drawing samples and this can go wrong in several problem instances. The idea can be engineered based on an application.

BAYESIAN UPDATES: Almost always the update procedure in filtering techniques rests on Bayes rule.

A.2 Robot pose learning (localization)

Pose of a robot is it's position and it's heading direction (x, y, θ). From now on pose is represented by using only x.

The online learning of the pose is posed as the following filtering problem:

$$p(x_t|z_{1:t}, u_{1:t}, m) \tag{A.3}$$

where

- $u_{1:t}$ is the sequence of odometry measurements.
- $z_{1:t}$ is the sequence of range measurements.
- m is the map of the environment the robot is in.

Using Bayes rule and assumptions (1^{st} order Markov process, observational independence) to figure out the updates:

$$\begin{aligned} p(x_t|z_{1:t}, u_{1:t}, m) &= p(x_t|z_{1:t-1}, u_{1:t}, m, z_t) \\ &= \frac{p(z_t|x_t, z_{1:t-1}, u_{1:t}, m)p(x_t|z_{1:t-1}, u_{1:t}, m)}{p(z_t|z_{1:t-1}, u_{1:t}, m)} \\ &= \eta p(z_t|x_t, m) \int_{x_{t-1}} p(x_t|x_{t-1}, u_t, m)p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m)dx_{t-1} \end{aligned} \tag{A.4}$$

NOW where comes the MC estimation? Thinking as an engineer only suffices :) As you can see we need to estimate the integral. We have the integral approximated by previous stage MC-estimation of $p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m)$ and hence can approximate the integral with the weighted kernel estimate as:

$$\int_{x_{t-1}} p(x_t|x_{t-1}, u_t, m)p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m)dx_{t-1} \approx \sum_{i=1}^{N} p(x_t|x_{t-1}^{(i)}, u_t, m)w_{t-1}^{(i)} \tag{A.5}$$

OK! We have samples for $p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m)$ but what about samples for $p(x_t|z_{1:t}, u_{1:t}, m)$. Since we cannot sample from $p(x_t|z_{1:t}, u_{1:t}, m)$ directly (if we could then there's no need for this "predict-update" problem) let's use $\int_{x_{t-1}} p(x_t|x_{t-1}, u_t, m)p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m)dx_{t-1}$ as the "proposal" and weight the particles as

$$\hat{w}_t(x_t) = \frac{p(x_t|z_{1:t}, u_{1:t}, m)}{\int_{x_{t-1}} p(x_t|x_{t-1}, u_t, m)p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m)dx_{t-1}} \tag{A.6}$$

$$= \frac{\eta p(z_t|x_t, m)\cancel{\int_{x_{t-1}} p(x_t|x_{t-1}, u_t, m)p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m)dx_{t-1}}}{\cancel{\int_{x_{t-1}} p(x_t|x_{t-1}, u_t, m)p(x_{t-1}|z_{1:t-1}, u_{1:t-1}, m)dx_{t-1}}} \tag{A.7}$$

Guess what, we just saw an example of *Sequential* MC estimation! It is iterative but we get *new* observations in every stage that let's us improve our estimate. Contrast this with Metropolis-Hastings (on your own) if interested.

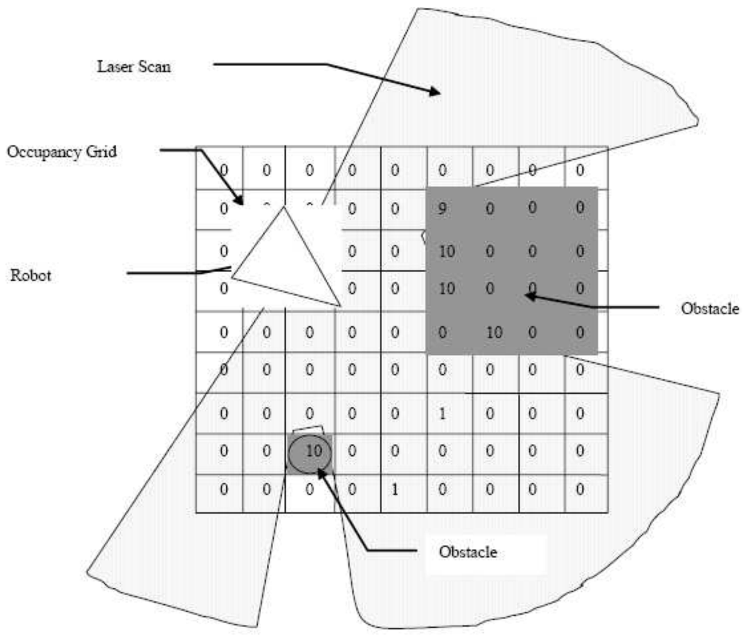


Figure 4 Occupancy grid after 10 scans

ALGORITHMIC PERSPECTIVE:

- **Initializing:** For $i = 1, \ldots, N$ draw $x_1^{(i)}$ from $p(x_1)$.
- **Sampling/Predicting:** For $i = 1, \ldots, N$ sample from the proposal $x_t^{(i)} \sim \sum_{j=1}^{N} p(x_t|x_{t-1}^{(j)}, u_t, m)w_{t-1}^{(j)}$.
- **Weighting/Updating:** For $i = 1, \ldots, N$ evaluate the importance weights $\hat{w}_t^{(i)} = \eta p(z_t|x_t^{(i)}, m)$. Then normalize the importance weights $w_t^{(i)} = \frac{\hat{w}_t^{(i)}}{\sum_{j=1}^{N} \hat{w}_t^{(j)}}$.

A.3 Robot map learning (mapping)

Here the state to be learnt is the map of the environment the robot is in. For an online learning the filter would be

$$p(m_t|x_{1:t}, z_{1:t}) \tag{A.8}$$

Guess what we have analytical solution for this filter!!! Occupancy grids is one example. We don't need MC-estimation. It would not be wise to use MC anyways since m_t is usually very high dimensional (equal to number of features in a map) and particle filter also suffers *curse of dimensionality*. Which says in a high dimensional space we need lot of data reason something strong because of the exponential increase in volume of the hypercube.

A.4 Simultaneous Localization and Mapping

The filter here is:

$$p(x_{1:t}, m_t | z_{1:t}, u_{1:t}) \tag{A.9}$$

For this there is no known analytic expression so we use MC estimation. So usually it's hard to filter more than ~ 50 features in m_t. But fortunately Rao-Blackwellization with state-decomposition theorem which roughly stating leads that if a state can be decomposed into analytical sub-parts then it's indeed *better* than direct estimation!

Since we know $p(m_t|x_{1:t}, z_{1:t})$ is analytically computable let's decompose the SLAM posterior as:

$$\begin{aligned} p(x_{1:t}, m_t|z_{1:t}, u_{1:t}) &= p(x_{1:t}|z_{1:t}, u_{1:t})p(m_t|z_{1:t}, u_{1:t}, x_{1:t}) \\ &= p(x_{1:t}|z_{1:t}, u_{1:t})p(m_t|z_{1:t}, x_{1:t}) \; (\because \; m_t \text{ is independent of } u_{1:t} \text{ given } z_{1:t}, x_{1:t}) \end{aligned} \tag{A.10}$$

So we need to track only $p(x_{1:t}|z_{1:t}, u_{1:t})$ using SMC estimation. Let's follow the approach we took for pose learning. Let's first get the update equation using Bayes rule:

$$p(x_{1:t}|z_{1:t}, u_{1:t}) = \eta p(z_t|z_{1:t-1}, x_{1:t}, u_{1:t})p(x_t|x_{1:t-1}, u_{1:t})p(x_{1:t-1}|z_{1:t-1}, u_{1:t-1}) \tag{A.11}$$

The derivation follows:

$$\begin{aligned} p(x_{1:t}|z_{1:t}, u_{1:t}) &= \frac{p(x_{1:t}|u_{1:t})p(z_{1:t}|x_{1:t}, u_{1:t})}{p(z_{1:t}|u_{1:t})} \\ &\left(\text{using Baye's rule } p(A|B, C) = \frac{p(A|B)p(C|A, B)}{p(C|B)}\right) \\ &\text{where } A = x_{1:t}, B = u_{1:t}, C = z_{1:t} \\ &= \frac{p(x_t|x_{1:t-1}, u_{1:t}) \, \boxed{p(x_{1:t-1}|u_{1:t-1})} \, p(z_t|z_{1:t-1}, x_{1:t}, u_{1:t}) \, \boxed{p(z_{1:t-1}|x_{1:t-1}, u_{1:t-1})}}{p(z_t|z_{1:t-1}, u_{1:t}) \, \boxed{p(z_{1:t-1}|u_{1:t-1})}} \\ &= \frac{p(x_t|x_{1:t-1}, u_{1:t})p(z_t|z_{1:t-1}, x_{1:t}, u_{1:t}) \, \boxed{p(x_{1:t-1}|z_{1:t-1}, u_{1:t-1})}}{p(z_t|z_{1:t-1}, u_{1:t})} \\ &\text{using Bayes rule with } A = x_{1:t-1}, B = u_{1:t-1}, C = z_{1:t-1} \\ &= \eta p(z_t|z_{1:t-1}, x_{1:t}, u_{1:t})p(x_t|x_{1:t-1}, u_{1:t})p(x_{1:t-1}|z_{1:t-1}, u_{1:t-1}) \end{aligned} \tag{A.12}$$

Contrast this with pose learning (eq. (A.4)) where we have an additional integral.

Since sampling directly from $p(x_{1:t}|z_{1:t}, u_{1:t})$ is hard we sample from a proposal π that is constructed *sequentially* as:

$$\pi(x_{1:t}|z_{1:t}, u_{1:t}) = \pi(x_t|x_{1:t-1}, z_{1:t}, u_{1:t})\pi(x_{1:t-1}|z_{1:t-1}, u_{1:t-1}) \tag{A.13}$$

which implies that $x_{1:t} \sim \pi(x_{1:t}|z_{1:t}, u_{1:t})$ actually means $x_t \sim \pi(x_t|x_{1:t-1}, z_{1:t}, u_{1:t})$ and $x_{1:t} =< x_t, x_{1:t-1} >$.

The important weights are computed as:

$$\begin{aligned}\hat{w}_t(x_{1:t}) &= \frac{p(x_{1:t}|z_{1:t},u_{1:t})}{\pi(x_{1:t}|z_{1:t},u_{1:t})}\\ &= \frac{\eta p(z_t|z_{1:t-1},x_{1:t},u_{1:t})p(x_t|x_{1:t-1},u_{1:t})\, p(x_{1:t-1}|z_{1:t-1},u_{1:t-1})}{\pi(x_t|x_{1:t-1},z_{1:t},u_{1:t})\, \pi(x_{1:t-1}|z_{1:t-1},u_{1:t-1})}\\ &= \hat{w}_{t-1}(x_{1:t-1})\frac{\eta p(z_t|z_{1:t-1},x_{1:t},u_{1:t})p(x_t|x_{1:t-1},u_{1:t})}{\pi(x_t|x_{1:t-1},z_{1:t},u_{1:t})} \end{aligned} \tag{A.14}$$

ALGORITHMIC PERSPECTIVE:

- **Initializing:** For $i = 1,\dots,N$, draw $x_1^{(i)}$ from $p(x_1)$.
- **Sampling/Predicting:** For $i = 1,\dots,N$, sample from the proposal $x_t^{(i)} \sim \pi(x_t|x_{1:t-1}^{(i)},z_{1:t},u_{1:t})$. And $x_{1:t}^{(i)} \equiv < x_t^{(i)}, x_{1:t-1}^{(i)} >$.
- **Weighting/Updating:** For $i = 1,\dots,N$, evaluate the importance weights $\hat{w}_t^{(i)} = \hat{w}_{t-1}^{(i)}\frac{\eta p(z_t|z_{1:t-1},x_{1:t}^{(i)},u_{1:t})p(x_t^{(i)}|x_{1:t-1}^{(i)},u_{1:t})}{\pi(x_t^{(i)}|x_{1:t-1}^{(i)},z_{1:t},u_{1:t})}$ Then normalize the importance weights $w_t^{(i)} = \frac{\hat{w}_t^{(i)}}{\sum_{j=1}^N \hat{w}_t^{(j)}}$.

A.5 Optimality, resampling schedule and proposal

OPTIMALITY: Optimality of filtering via simulation can be measured using variance of the importance weights of the particles. In a perfect simulation scenario, weights of all particles must be same meaning the samples are *randomly* drawn from the posterior which is being simulated. This can also be noted in the weight equations above where the importance weights would be 1 if $\pi = p$ and the normalized weights would be $1/N$. In such a case the variance of weights will be 0.

But it's been shown that the variance increases over time for a finite N. This problem is called *weight degeneracy* problem.

RESAMPLING: Resampling helps in reducing the variance by replacing low weight particles with duplicates of higher weight particles. But naive resampling might lead to having only one particle duplicated which is called *particle depletion* and if you notice this also means the particles are not random in fact they are too deterministic. One important note is that after resampling, the weights of all particles are reset to $1/N$.

Adaptive resampling is an easy engineering fix that let's us schedule resampling according to a flag that tries to maintain a balance between reducing variance and avoiding bias.

The flag variable is $N_{eff} = \frac{1}{\sum_{i=1}^N w_t^{(i)}}$. If $N_{eff} < N/2$ resampling is *not* done otherwise we resample.

There are more complex fixes available.

PROPOSAL DISTRIBUTION: Though resampling is a temporary relief it is not a *solution*. Selection and design of proposal distribution can be helpful in spreading around particles into low variance and high likelihood regions. But in general there is no restriction from the particle filter point of view to choose a proposal distribution. We will look at two of them viz.

1. $p(x_t|x_{1:t-1}, u_{1:t})$
2. $p(x_t|x_{1:t-1}, z_{1:t}, u_{1:t})$

If we choose the first option then besides drawing samples the main difference is in the weight recursion eq. (A.14).

- **Sampling:** Drawing $x_t \sim p(x_t|x_{1:t-1}, u_{1:t})$ is simple because we are *given* a closed form which is usually Gaussian of the form $\mathcal{N}(\mu, \Sigma)$ based on the odometry motion model of the robot.
- **Weight update:** By plugging in the proposal eq. (A.14) is simplified as:

$$\begin{aligned} w_t(x_{1:t}) &= \eta w_{t-1}(x_{1:t-1}) \frac{\cancel{p(x_t|x_{1:t-1}, u_{1:t})} p(z_t|z_{1:t-1}, x_{1:t}, u_{1:t})}{\cancel{p(x_t|x_{1:t-1}, u_{1:t})}} \\ &= \eta w_{t-1}(x_{1:t-1}) p(z_t|z_{1:t-1}, x_{1:t}, u_{1:t}) \\ &= \eta w_{t-1}(x_{1:t-1}) p(z_t|m_{t-1}, x_t) \because \ z_t \text{ is independent of } u_{1:t} \end{aligned} \tag{A.15}$$

and conditioning on $x_{1:t-1}, z_{1:t-1}$ is equivalent to conditioning on m_{t-1}.

If we choose the second option then sampling is also non-trivial because there is no easy closed form available. So we use MC-estimation to simulate the proposal distribution!

So let's get the update equation for the optimal proposal.

$$\begin{aligned} p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t}) &= p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1}, z_t) \\ &= \frac{p(z_t|x_t, x_{1:t-1}, u_{1:t}, z_{1:t-1}) p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})}{p(z_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})} \\ & \left(\text{using Bayes rule } p(A|B, C, D, E) = \frac{p(E|A, B, C, D) p(A|B, C, D)}{p(E|B, C, D)} \right) \\ & \text{where } A = x_t, B = x_{1:t-1}, C = u_{1:t}, D = z_{1:t-1}, E = z_t \\ &= \frac{p(z_t|x_t, x_{1:t-1}, u_{1:t}, z_{1:t-1}) p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})}{\int_{x_t} p(z_t|x_t, x_{1:t-1}, u_{1:t}, z_{1:t-1}) p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1}) dx_t} \\ & (\text{marginalizing over } x_t) \end{aligned} \tag{A.16}$$

Since we cannot directly sample from the optimal proposal let's sample from $p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})$ and assign weights (CAUTION: these are called first-stage weights and not to be confused with the actual weights for the particles representing the state $x_{1:t}^{(i)}$) to be:

$$\begin{aligned}
\tilde{w}(x_t) &= \frac{p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t})}{p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})} \\
&= \frac{\frac{p(z_t|x_t,x_{1:t-1},u_{1:t},z_{1:t-1})\cancel{p(x_t|x_{1:t-1},u_{1:t},z_{1:t-1})}}{\int_{x_t} p(z_t|x_t,x_{1:t-1},u_{1:t},z_{1:t-1})p(x_t|x_{1:t-1},u_{1:t},z_{1:t-1})dx_t}}{\cancel{p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})}} \\
&= \frac{p(z_t|x_t, x_{1:t-1}, u_{1:t}, z_{1:t-1})}{\int_{x_t} p(z_t|x_t, x_{1:t-1}, u_{1:t}, z_{1:t-1})p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})dx_t} \\
&\approx \frac{p(z_t|x_t, x_{1:t-1}, z_{1:t-1}, u_{1:t})}{\frac{1}{K}\sum_{j=1}^{K} p(z_t|\tilde{x}_t^{(j)}, x_{1:t-1}, z_{1:t-1}, u_{1:t})} \\
&= \frac{p(z_t|x_t, m_{t-1})}{\frac{1}{K}\sum_{j=1}^{K} p(z_t|\tilde{x}_t^{(j)}, m_{t-1})} \qquad \text{(A.17)}
\end{aligned}$$

where $\{\tilde{x}_t^{(j)}\}_{j=1}^{K}$ are the K samples drawn from $p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})$. And conditioning $x_{1:t-1}, z_{1:t-1}$ is equivalent to conditioning on m_{t-1}. And z_t is independent of $u_{1:t}$. Now let's look at the **weight update**.

$$\begin{aligned}
w_t(x_{1:t}) &= w_{t-1}(x_{1:t-1})\frac{\eta p(x_t|x_{1:t-1}, u_{1:t})p(z_t|z_{1:t-1}, x_{1:t}, u_{1:t})}{p(x_t|x_{1:t-1}, z_{1:t}, u_{1:t})} \\
&= w_{t-1}(x_{1:t-1})\frac{\eta p(x_t|x_{1:t-1}, u_{1:t})p(z_t|z_{1:t-1}, x_{1:t}, u_{1:t})}{p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1}, z_t)} \\
&= w_{t-1}(x_{1:t-1})\frac{\eta p(x_t|x_{1:t-1}, u_{1:t})p(z_t|z_{1:t-1}, x_{1:t}, u_{1:t})}{\frac{p(z_t|x_t,x_{1:t-1},u_{1:t},z_{1:t-1})p(x_t|x_{1:t-1},u_{1:t},z_{1:t-1})}{p(z_t|x_{1:t-1},u_{1:t},z_{1:t-1})}} \text{ using Bayes rule} \\
&= w_{t-1}(x_{1:t-1})\frac{\eta p(x_t|x_{1:t-1}, u_{1:t})\cancel{p(z_t|z_{1:t-1}, x_{1:t}, u_{1:t})}}{\frac{\cancel{p(z_t|x_t,x_{1:t-1},u_{1:t},z_{1:t-1})}p(x_t|x_{1:t-1},u_{1:t},z_{1:t-1})}{p(z_t|x_{1:t-1},u_{1:t},z_{1:t-1})}} \\
&= w_{t-1}(x_{1:t-1})\frac{\eta p(x_t|x_{1:t-1}, u_{1:t})}{p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})} p(z_t|x_{1:t-1}, u_{1:t}, z_{1:t-1}) \\
&= w_{t-1}(x_{1:t-1})\frac{p(x_t|x_{1:t-1}, u_{1:t})}{p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})} \int_{x_t} p(z_t|x_t, x_{1:t-1}, u_{1:t}, z_{1:t-1})p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})dx_t \\
&\approx w_{t-1}(x_{1:t-1})\frac{p(x_t|x_{1:t-1}, u_{1:t})}{p(x_t|x_{1:t-1}, u_{1:t}, z_{1:t-1})} \frac{1}{K}\sum_{j=1}^{K} p(z_t|\tilde{x}_t^{(j)}, x_{1:t-1}, z_{1:t-1}, u_{1:t}) \qquad \text{(A.18)}
\end{aligned}$$

A.6 Caveats to be aware of

Filtering SLAM posterior is significantly more involved compared to Localization. Most important catch is that we sample in an *increasing* space of dimension with time and it is *bound* to diverge!!! There's already work done to fix this to some extent using "Marginal Particle Filters" which filters in the fixed dimension only. The key contributions by those guys was reducing the SLAM posterior to similar to that of Localization *and* also clever computational speedups.