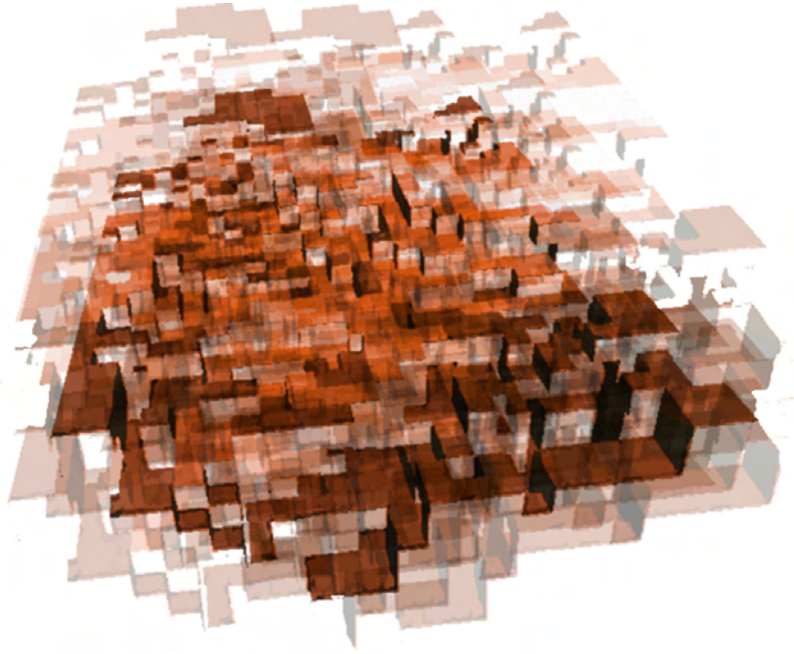
Autonomous Robot Work Cell Exploration using Multisensory Eye-in-Hand Systems



Von der Fakultät für Maschinenbau
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des akademischen Grades
Doktor-Ingenieur
genehmigte
Dissertation
von

Dipl.-Ing. Michael Suppa

geboren am 13.01.1976 in Hannover

DLR ifR

2008

1. Referent: Prof. Dr.-Ing. Bodo Heimann
2. Referent: Prof. Dr.-Ing. Gerd Hirzinger

3. Referent: Prof. Dr. Gregory Hager

Vorsitzender der Prüfungskommission: Prof. Dr.-Ing. Berend Denkena

Tag der Promotion: 19.11.2007

Preface

This Ph.D. Thesis was written during my employment at the Institute of Robotics and Mechatronics at the German Aerospace Center (DLR) in Oberpfaffenhofen, Germany. First of all, I would like to thank the Head of this Institute, Prof. Gerd Hirzinger, for the opportunity to work in the field of 3-D modeling and exploration - always a challenge, yet the best conceivable work environment for pursuing my objectives and advancing my ideas.

I would like to thank the supervisor of this Ph.D. Thesis, Prof. Bodo Heimann (Institute for Robotics, University of Hannover, Germany), for his continuous support and advice. Prof. Kamal Gupta (Simon Fraser University, Burnaby, Canada) permitted me as a visiting researcher to his laboratory, granted me access to an excellent 2-D planning environment, and helped evolve numerous ideas and thought-provoking impulses in extensive discussions. Further, I extend my thanks to Prof. Greg Hager (Johns Hopkins University, Baltimore, USA) for valuable advice and affirmative reassurance.

Many thanks to my colleagues of the 3-D Modeling Group at DLR for good teamwork and interesting, far-ranging discussions. Special thanks go to Tim Bodenmueller, Rainer Konietschke, Wolfgang Sepp and Klaus Strobl for dedicated support and proofreading, Prof. Darius Burschka for his advice and considerable time spent in discussions with me, Christian Rink for an honest opinion, support, and proofreading, Simon Kielhoefer for brilliant mechanical constructions, and Tilo Wuesthoff for design support. Many thanks to the service team.

During internships and master's theses, Robert Burger, Stefan Dietl, Tino Puegner, and Kevin Yoon have performed skilled experiments and implementations that have become substantial elements of this thesis - I am grateful for these contributions.

Many discussions and sometimes controversial arguments with Pengpeng Wang led towards important insights and food for thought. These inspirations are greatly appreciated.

I owe thanks to my friends and family, who encouraged me in finishing this thesis, never failing to believe in my abilities and motivating me. Moreover, I would like to thank Mareike Doepke for an essential impulse, rigorous support, and uncomplaining endurance of weekends full of work, as well as for repeated proofreading and the improvement of the thesis's language.

Last but not least, I would like to thank my mother, who passed away much too early, and father, Gudrun and Rainer-Ruediger Suppa. Without my parents, I would not be who I am.

Munich, September 2007 Michael Suppa

Abstract

The thesis *Autonomous Robot Work Cell Exploration using Multisensory Eye-in-Hand Systems* presents a sensor-based approach to the self-guided robotic exploration of initially partly unknown environments, which takes sensing uncertainty into account.

Aiming at facilitating automated work processes in flexible work cells, the robotic system is designed to gain the largest possible maneuverability, i.e. to maximize its knowledge of the configuration space. An efficient and reliable exploration of flexible work cells is performed by comprehensively integrating various fields of knowledge: flexible sensor systems, probabilistic environment representations that consider sensor uncertainty, view and motion planning designed to acquire a maximum amount of knowledge while using a minimal number of view points, and, lastly, the fusion of information based on an optimized update rule.

In an initially partly unknown environment a robot is tasked with incrementally establishing a sound representation of its surroundings, thus enabling it to securely perform task-directed exploration processes, e.g. object modeling in physical space. As it is crucial for safe motion planning to acquire reliable and secure information on obstacles, a multisensory system is applied. In a number of simultaneous sensor-based operations the physical space of the robot is gradually divided into regions which are defined either as *safe-for-motion* or as *planned-for-exploration*.

Main focus is set on sensors in hand-eye configuration and on robots with non-trivial kinematics, i.e. articulated robots with many degrees of freedom and complex geometries. Algorithms considering measurement noise and occlusion are developed for an efficient planning of Next-Best-Views.

Methods and sensors are evaluated in 3-D simulations; their results are evolved into design criteria for a multi-purpose exploration sensor, the 3D-Modeller. The sensor is implemented and evaluated in realistic experiments with a Kuka KR16 robot: Measurement is conducted in a test-bed which authentically represents a flexible work cell. Experiments for pure work space exploration, an exploration of pre-defined regions of interest in physical space, and combined missions are successfully performed based on different grid-based update rules. The developed method, considering environment uncertainty in the planning process, enables information gain-driven missions such as view planning for object recognition or grasp planning.

Keywords: autonomous exploration, eye-in-hand system, multisensory exploration, self-guided sensing, view planning, motion planning

Zusammenfassung

Die unter dem Titel *Autonomous Robot Work Cell Exploration using Multisensory Eye-in-Hand Systems - Selbständige Erkundung von Roboterarbeitsräumen unter Einsatz multisensorieller Hand-Auge-Systeme* in englischer Sprache verfasste Doktorarbeit stellt ein neuartiges Verfahren zur eigenständigen Untersuchung von teilweise unbekannten Umgebungen durch Roboter vor, welches mögliche Messunsicherheiten berücksichtigt.

Ein Robotersystem wird in die Lage versetzt, schrittweise ein zuverlässiges Modell seiner zunächst teilweise unbekannten Umwelt zu erzeugen. Dieses Modell dient als Basis für eine sichere Planung und Durchführung von Aufgaben wie z.B. Objektmodellierung. Zielsetzung des Ansatzes ist es, automatisierte Arbeitsprozesse in Roboterarbeitsräumen zu vereinfachen. Das Robotersystem ist darauf ausgelegt, sich die größtmögliche Bewegungsfähigkeit zu schaffen, d.h. die Kenntnis seines Konfigurationsraumes effizient zu maximieren.

Die zuverlässige Erfassung von Umweltinformationen ist von zentraler Bedeutung für eine sichere Bewegungs- und Aufgabenplanung. Daher wird ein System verwendet, in welchem mehrere verschiedenartige Sensoren simultan die Umgebung erfassen. So kann die physikalische Umgebung des Roboters stufenweise in Gebiete eingeteilt werden, die entweder als *safe-for-motion* (sichere Bewegungen sind möglich) oder *planned-for-exploration* (zur Erkundung vorgesehen) definiert werden.

Eine Kombination unterschiedlicher Wissensgebiete gewährleistet die Zuverlässigkeit der Erkundungsprozesse: flexible Sensorsysteme, wahrscheinlichkeitstheoretische Umweltmodellierungen, optimierte Aufnahme- und Bewegungsplanungen (mit maximalem Informationsgewinn bei minimalem Messaufwand), sowie nicht zuletzt eine gezielte Fusion der Informationen.

Das Verfahren ist auf Roboter mit nicht-trivialer Kinematik, d.h. solche mit zahlreichen Freiheitsgraden und komplexen geometrischen Eigenschaften, ausgelegt die mit so genannten Hand-Auge-Sensorsystemen ausgestattet sind.

Im Rahmen des Erkundungsprozesses werden verschiedene Entfernungsmesswerte fusioniert, um freie Bereiche eines Arbeitsraumes zuverlässig bestimmen zu können. Zudem werden Algorithmen zur Bestimmung der optimalen Sensorlage für die jeweils nächste Messung (sog. Next-Best-Views) unter Berücksichtigung möglicher Messfehler sowie der eventuellen Verdeckung einzelner Bereiche entwickelt.

Die Vorgehensweise sowie die ausgewählten Sensoren werden zunächst im Rahmen von 3-D-Simulationen ausgewertet. Aus den Simulationsergebnissen wird die detaillierte Spezifikation für

einen vielseitig anwendbaren Explorationssensor abgeleitet: Der 3D-Modellierer wird implementiert und in wirklichkeitsnahen Experimenten an einem Kuka KR16-Roboter evaluiert.

Die Testumgebung stellt ein realistisches Abbild einer typischen flexiblen Roboterarbeitszelle dar. Experimente zur Exploration des eigentlichen Arbeitsraumes und vorab definierter Explorationsvolumina sowie kombinierte Aufgaben werden erfolgreich durchgeführt. Dabei werden verschiedene Aufdatierungen des gitterbasierten Umweltmodells angewendet. Die entwickelte Explorationsstrategie berücksichtigt die Unsicherheit der Roboterumgebung während der Planungsphase. Sie ermöglicht verschiedene Missionen wie z.B. Objekterkennung oder Greifplanung, bei denen die unvollständige Umweltinformation durch Exploration gewonnen werden kann.

Schlagworte: Autonome Exploration, Hand-Auge System, Multisensorielles Messen, Planung von Sensorlagen, Bewegungsplanung

Contents

List of Figures

List of Tables

Abbreviations and Variables

Abbreviations

AABB	Axis-Aligned Bounding Box
AOD	Adaptive Obstacle Density Method
AR	Augmented Reality
BNRS	Beam Noise Roadmap Sweep view planning method
BRS	Beam Roadmap Sweep view planning method
CAN	Controller Area Network: priority-based field bus with real-time capabilities
CCD	Charged-Coupled Device
CD	Collision Detection
CT	Computer Tomography
DoF	Degree of Freedom
DS	Dempster-Shafer
DH	Denavit-Hartenberg
EM	Expectation Maximization
GJK	Gilbert-Johnson-Keerthi
GPS	Global Positioning System
iff	if and only if
IG	Information Gain
ISA	Incremental Separating-Axis
KRL	Kuka Robot Language: High-level programming language for Kuka robot programs
LRS	Laser-Range Scanner
LSP	Laser-Stripe Profiler

MER	Maximum Entropy Reduction
MAP	Maximum a Posterior
MBS	Multi-Body-System
MPVRS	Maximum Physical Volume Sweep view planning method
MWMF	Multiple Window Multiple Filter
NBV	Next-Best-View
NP	Non-deterministic Polynomial time
OBB	Oriented Bounding Box
P	Point view planning method
PBS	Point Beam roadmap Sweep view planning method
PDF	Probability Density Function
PRM	Probabilistic RoadMap
PSTS	PhysicalSphereTreeSpace
RoI	Region of Interest
RPC	Remote Procedure Call
SCP	Set Covering Problem
SCS	Stereo Camera Sensor
SGM	Semi Global Matching
SLAM	Simultaneous Localization And Mapping
SME	Small and Medium-sized Enterprises
TCP	Tool Center Point
UDP	User Datagram Protocol
UML	Unified Modeling Language
VFF	Virtual Force Field
VPP	View Planning Problem
VRML	Virtual Reality Modeling Language

Variables

$\mathcal{A}(\mathbf{q})$	Volume in $\mathcal{P}$-space occupied by robot in configuration $\mathbf{q}$
$\mathcal{C}$	Symbol for the Configuration Space

$\mathcal{C}_{free}$	Known free part of the Configuration Space
$\mathcal{C}_{known}$	Known part of the Configuration Space
$\mathcal{C}_{occ}$	Known occupied part of the Configuration Space
$\mathcal{C}_{unk}$	Unknown part of the Configuration Space
$\partial\mathcal{C}$	Boundary of $\mathcal{C}$
c_i	Cell of with index i of the physical space map
$\mathcal{C}$-space	Configuration Space, i.e. Planning Space $\mathcal{C} \in \mathbb{R}^N$, where N denotes the Degrees of Freedom of the robot, i.e. joints.
d	Distance measurement; $d \in [d_{min}, d_{max}]$.
d_{c_i}	Distance of the cell c_i from the sensor origin
F_i	Symbol for coordinate frames, the subscript i names/indexes the coordinate system/frame.
$H(\mathcal{C})$	$\mathcal{C}$-space entropy
$\mathcal{I}$	Information Gain (IG) map, defined in $\mathcal{P}$-space
λ	Density of obstacle distribution for a Poisson Point Process, $\lambda \in]0, \infty[$
μ	Mean of a measurement
ν	Maximal induced belief in the forward sensor model for Fuzzy and Belief Update
m	Map of the environment
$\mathcal{L}$	Robot motion path, i.e. edges in $\mathcal{R}$
$\mathcal{P}$	Symbol for the Physical Space
$\mathcal{P}_{free}$	Known free part of the Physical Space
$\mathcal{P}_{known}$	Known part of the Physical Space
$\mathcal{P}_{occ}$	Known occupied part of the Physical Space
$\mathcal{P}_{unk}$	Unknown part of the Physical Space
$\mathcal{P}$-space	Physical Space, i.e. Cartesian Space, $\mathcal{P} \in \mathbb{R}^3$. $\mathcal{P}$ is assumed to consist of axis aligned primitives (voxels, spheres) unless otherwise stated, therefore $\mathbb{R}^6$ maps to $\mathbb{R}^3$.
$\mathbf{q}$	Configuration in $\mathcal{C}$-space

$\mathcal{R}$	Roadmap, describing the connectivity of $\mathcal{C}_{free}$. The roadmap is a graph, whose edges are collision-free paths $\mathcal{L}$ and its nodes are free configurations $\mathbf{q} \in \mathcal{C}_{free}$.
r_{IG}	$\mathcal{I}$-space resolution
r_P	$\mathcal{P}$-space resolution
σ^2	Variance of a measurement
s_{max}	Optimal measurement regarding the planning criterion, i.e. the NBV.
s	Sensor configuration, the optimal sensing is denoted s_{max}.
$\mathcal{T}$-space	Task Space, a task-directed subset of the work space $\mathcal{C}$ or physical space $\mathcal{P}$ depending on the task-definition space, e.g. a region of interest for object recognition is a $\mathcal{T}$-space subset of $\mathcal{P}$.
u	Control variable, i.e. the path to the next best view.
$\mathcal{V}$	Sensor field of view in $\mathcal{P}$-space
$\mathbf{x}$	Pose of the robotic system in Cartesian coordinates, i.e. in $\mathcal{P}$-space.
$\mathbf{x}_s$	Sensor pose, i.e. the sensor origin at robot pose $\mathbf{x}$ or configuration $\mathbf{q}$, $\mathbf{x}_s = [\mathbf{x}_{loc}; \mathbf{x}_{dir}] \in \mathbb{R}^6$.
$\mathbf{x}_{dir}$	Vector denoting the sensor orientation, $\mathbf{x}_{dir} \in \mathbb{R}^3 \setminus \{0\}$
$\mathbf{x}_{loc}$	Vector denoting the sensor location, $\mathbf{x}_{loc} \in \mathbb{R}^3$
ζ	Sensor-dependent width of a region around the real measurement d in the forward sensor model.
z	Generalized measurement variable

1 Introduction

This thesis presents a sensor-based approach to the autonomous robotic exploration of initially partly unknown environments. As it is most crucial for a safe motion planning to acquire reliable and secure information about obstacles, a multisensory system is applied, integrating several sensors and respecting each sensors specific characteristics. This system enables a robot to gain secure information and to efficiently fulfill the task of incrementally building a representation of its surroundings. A reliable work space representation is the essential basis for a secure performance of task-directed exploration and inspection processes. Tasks such as object tracking, object recognition, or object modeling are supported by the system likewise.

The following sections provide an introduction to the thesis: The Problem Statement addresses the manifold and versatile challenges and restrictions a robotic system faces, usually in parallel, when tasked with safely exploring its environment. Current shortcomings are addressed; interrelated modules required for solving the autonomous exploration problem are discussed. Further, the Contribution of the Thesis, i.e. the significant advancements this work offers for safe motion planning and exploration processes in robotics, is presented. Concluding, the Outline of the Thesis is exposed, offering an insight into approach and structure of the document.

This work has been partially funded by the European Commissions Sixth Framework Programme under Grant No. 011838 as part of the Integrated Project SMErobot™.

1.1 Problem Statement

In service robotics, the robot's environment is usually cluttered and unstructured. Motion planning in several Degrees of Freedom (DoF), planning of tasks such as grasping, or object recognition must be performed by the robot. The surroundings are usually only partially known; thus exploration strategies are required. Additionally, moving objects and interacting humans are part of the environment. In case of a mobile system, localization and motion uncertainty must also be considered.

In industrial robotics, small-batch production with direct human-robot interaction is an emerging field of application. As work cells must be flexible, the nominal model needs to be adapted. This is usually not done in current applications of mass production, e.g. in automotive industry. In case of robots being used in Small and Medium-sized Enterprises (SME), the space available for a robot cell is often very limited; the cell cannot be completely secured and closed off by fences. The robot must be able to adapt to changing tasks and environments, requiring an autonomous sensor-based exploration of the work cell.

The field of (sensor-based) motion planning rarely considers the possibilities of multisensory data acquisition. Two main reasons are the lack of appropriate systems and the popular assumption of idealized sensing prevailing in the motion planning community today. The potential impact of multisensory information has been assessed with a system acquiring 3-D information about the environment in hand-guided operation [154], showing the performance of fused range data for building high-quality 3-D models of small-scale objects. The experience gained in the development of this manually actuated system led to the idea of automating the process of digitizing objects. These efforts resulted in a robotic system capable of actively performing automated inspection tasks in partially known 3-D environments. The concept incorporates sensor noise models, as every real-life sensing system is subject to noise. This enables the robotic system to autonomously[1] determine further actions to be taken.

As the robot is moving in a priori unknown environments, it is fundamental for all tasks that the robotic system gains the largest possible maneuverability. The system must maximize the knowledge of its configuration space, denoted $\mathcal{C}$-space, aiming at robust motion in the known free space.

[1]The term autonomy, derived from Greek (Auto-Nomos- nomos meaning "law": One who gives oneself his own law) as used in philosophy implies the capacity of a rational individual to informedly and uncoercedly decide. Given this definition, current robotic systems are much closer to automatic (=self operating) machines than to autonomous machines. While the terms automatic and autonomous, are commonly used synonymously in robotics, in this work the decision on optimal view planning is derived from an autonomous procedure in the true sense of the word.

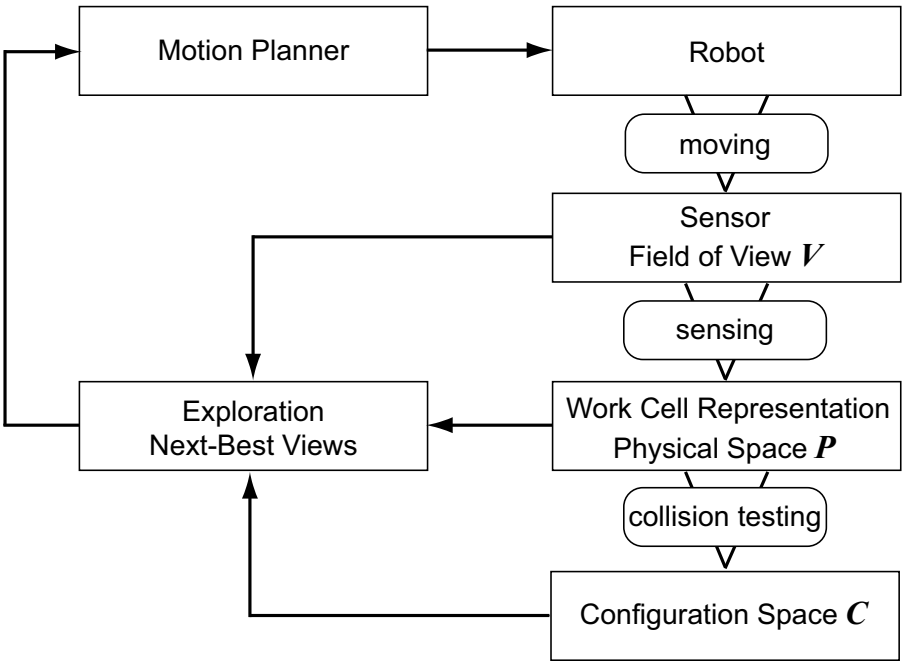


Figure 1.1: Interrelation of modules required for solving the autonomous exploration problem.

In order to increase robustness, multisensory range information is used. It must never happen that an occupied region is erroneously sensed free[2], e.g. because opaque objects are not detected by a laser, and defined as free for motion planning. Therefore, the next motion or sensing operation is inferred based on the respective sensor model, which also allows updating of the environment after a measurement was performed.

Exploration with robots featuring complex kinematics requires research on multiple topics. The interrelation of the required modules is presented in Fig. 1.1. The robot, described by its geometry and kinematics and equipped with several sensors, acquires a partially unknown environment, the physical space $\mathcal{P}$. The sensors can be range sensors, skin sensors, touch sensors, or force sensors. In this thesis, the field of sensors is limited to non-contact sensors, as the interaction of standard industrial robots with unknown environment is best performed contactless. The navigation is done in configuration space $\mathcal{C}$, its connectivity is stored in a roadmap. The $\mathcal{C}$-space is mapped to $\mathcal{P}$-space by the forward kinematics and geometry of the robot, while $\mathcal{C}_{free}$ is obtained through collision tests. Exploration is performed by computing optimal sensings, so-called Next-Best-Views (NBV); the motion planner computes the path to the desired NBV.

[2]This event is also called false true detection case

In the following, topics relevant for exploration, as depicted in Fig. 1.1, are summarized, focusing on sensing and physical space representations, motion and view planning, and sensor technology:

Multisensory fusion in environment representation: In pure motion planning problems, the environment is represented in two states, *occupied* (occ) or *free*. The approach assumes the environment to be completely known. In exploration, i.e. sensor-based motion planning, *unknown* (unk) areas must be represented; their exploration must be aspired. In sensor-based motion planning, sensing is commonly assumed to be ideal in the planning stage and during task execution. As sensing is never ideal in reality, uncertainty of sensing requires consideration in sensor-based operation; therefore the state of the environment must range continuously from *free* to *occupied*. For motion planning, a threshold for a *free* region must be defined. A well-known approach in 2-D physical spaces, common for mobile robots, is the use of occupancy grids. The environment is tesselated assigning a continuous state for each cell. However, implementations in 3-D environments are rare. The representation must be scalable in order to account for large environments as well as narrow passages. As sensors are prone to errors induced by surface properties, multiple sensors must be used in order to reliably define free regions.

The information acquired by multiple sensors must be fused into an environment representation. Bayes', Belief, and Fuzzy Update are three main approaches to fuse information, each being beneficial if applied properly. They are compared with the Naïve Update, which corresponds to ideal sensing without uncertainty. Bayes' Update is the most common approach in robotics. It applies an inverse sensor model, requiring independence of the measurements. While complete ignorance cannot be modeled by Bayes' Rule, Belief Theory is able to address this property, still requiring independence. Belief models the hypothesis for *occupied* and *free* independently, allowing a determination of contradiction. Fuzzy Update is most controversially discussed in robotics. The independence of measurements is not required, yet operations must be formulated carefully, as it lacks probabilistic measures such as entropy. Information gain is required for driving the exploration, therefore, this thesis defines the information gain for Fuzzy Update by computing a grey value for an occupancy state based on the *free* and *occupied* Fuzzy sets.

Motion and view planning: The robot must move safely in the work space that is known to be free. For large work spaces, sparse representations of $\mathcal{C}$-space must allow a memory-efficient planning. As the known portion of the environment changes with time and information gain, an incremental and extendable representation for navigation

is required. Sensor-based motion planning involves a view planning component. Common approaches range from random sampling to sophisticated view planning considering sensor characteristics. This is commonly denoted computation of NBV and performed in order to gain as much knowledge as possible w.r.t the optimization criterion. Common goals are an efficient exploration of work space $\mathcal{C}$ and task space $\mathcal{T}$, a subset of the physical space $\mathcal{P}$ or $\mathcal{C}$ w.r.t. the definition space of the task. Examples for such tasks are object inspection (defined in $\mathcal{P}$) or goal configuration exploration (defined in $\mathcal{C}$).

Sensor technology: Today's off-the-shelf sensors do not meet the requirements of multisensory exploration, therefore research in this field is necessary. Application needs regarding data processing are very diverse; exploration requires *safe-for-motion* areas, for modeling of objects, high-quality data is crucial. These requirements are solely met by a multi-purpose vision platform, a system capable of acquiring data (range and texture) in order to provide information for all major tasks in robotics; in service robotics as well as robot's flexible work cells.

The exploration problem of complex kinematic robots (as opposed to mobile robots) lacks a general approach which considers that the robot must acquire knowledge of its maneuverability, i.e. its work space, while performing tasks in physical space. These two exploration aspects must be fused in order to provide a measure for planning the views. Additionally, the requirements the sensor must meet are different for each task, e.g. modeling requires noiseless, accurate data to compute smooth surfaces, whereas exploration needs any information available on an obstacle to ensure safe motion. Therefore, most methods focus on one task to be performed at a time. Data fusion also increases safety for motion. Further, the world model representation must allow for different data abstraction levels, as not all tasks process the same information.

This section identifies the common problems of sensor-based exploration. In the next section, the contributions of the thesis addressing common shortcomings are presented.

1.2 Contribution of the Thesis

The field of sensor-based exploration of partially known environments under presence of non-ideal surfaces requires interdisciplinary knowledge and skill. This enables a system to act autonomously: mechatronics, computer graphics, computer vision, and motion planning, to name a few. Various approaches cover subsets of the entire problem. The work implemented in this thesis aims at an an integrated consolidation, as this has not yet been attempted:

- The approach (method and implementation) combines imprecise sensing in sensor-based exploration in physical space (under consideration of work space constraints) with a general view planning in physical space, verified in real experiments.
- The sensor data is fused in maps. A comparison of map update methods for 3-D occupancy grids in multi-scale representations is performed.
- Uncertain information on the physical space is used for weakening the ideal sensing assumption towards planning with uncertainty, while in computational geometry these methods assume ideal sensing.
- A general measure of information based on entropy is developed, serving as a task-specific driving force for exploration, fusing physical and work space exploration into a common framework.
- Design criteria for the development of versatile exploration sensors, enabling robots with complex geometry and kinematics to identify *safe-for-motion* regions under presence of uncertain sensing, are defined. Requirements for exploration of the work space in combination with task-specific goals are considered, resulting in the development of a multi-purpose vision platform. Thus, the autonomous operation of robots in flexible work cells is enabled.

These developments have been partially published at international conferences and in journals: Parts of the synchronization concept and calibration are published in [154] and [158]. Consideration of noise in the planning stage of motion planning in 2-D environments is presented in [159]. Multisensory exploration is adressed in [155]. The articles [156] and [157] present applications of the method in flexible work cells.

The following topics have not yet been published: comparison of different update rules, multi-scale work space representation, the influence of sampling strategies on the exploration, and fusion of work and configuration space exploration, i.e. the task-directed exploration problem.

1.3 Outline of the Thesis

The thesis is organized as follows: Chap. 2 of this thesis summarizes the current **State of the Art** regarding exploration, information fusion, view planning, and environment representations. Criteria for the evaluation of incremental methods for $\mathcal{C}$-space exploration and inspection are shown. Additionally, existing systems and methods for environment information acquisition are presented, demonstrating the need

for an exploration sensor. Chap. 2 concludes with a clear statement on the missing links in solving the exploration problem generally, respecting sensor uncertainty in work and planning space.

The general problem of **Exploration of Robot Work Space** is addressed in Chap. 3. A short overview on current entropy approaches to $\mathcal{C}$-space exploration is given. An entropy-based NBV approach using sensor uncertainty is developed. The approach is designed for exploring the planning space, i.e. $\mathcal{C}$-space, the knowledge of which is required for planning tasks in physical space $\mathcal{P}$. In addition, basic methods used throughout the thesis and the corresponding update rules for global maps are presented.

One main task performed during **Exploration of Physical Space** is the inspection of objects. Chap. 4 contains the theoretic basics for determining the NBV for exploring unknown objects. Since the models are a priori unknown, views are derived model-free.

Subsequently, the overall framework of a combined exploration and inspection approach enabling **Multiple Task Exploration** is described in Chap. 5. The extendability to object recognition and other tasks is demonstrated, providing an entropy-based framework for general information gain computation and view planning.

Simulation of Exploration Tasks are performed in order to evaluate the methods in a novel simulation environment, which is presented in Chap. 6. The design criteria and requirements for a versatile vision platform for exploration and inspection are derived. Based on these prerequisites, simulations using a newly developed 3-D simulation suite are performed. The design rules for a suitable sensing system are presented. Sampling parameters, map updates, and view planning methods - developed in this thesis and in related literature - are compared w.r.t. their applicability in a multisensory system.

The simulation results lead towards the development of the 3D-Modeller, a **Multisensory Eye-in-Hand System**. This system, described in Chap. 7, is used for verification of exploration and inspection methods. Remarks on the extendability towards object recognition and tracking close the chapter.

Experiments in Robot Work Cells prove the performance of the methods. The system setup as well as the results obtained in a SME-suitable work environment are described in Chap. 8. In this test-bed, the flexibility and reliability of the approach in a large work cell with multiple exploration targets is presented, demonstrating its potential for service robotics and small batch assembly in SMEs.

The thesis' closing Chap. 9 provides the derived **Conclusion and Perspective**, offering a summary and outlook on future work. Open issues are discussed; further, future potentials, highlighting the extendability of the approach to mobile robotics, medical and cultural heritage applications, are anticipated.

2
State of the Art

This chapter covers the current State of the Art of exploration using robots with complex geometry and kinematics, taking the related fields of research into account.

The general exploration problem is discussed. Mapping procedures and exploration strategies as well as methods for computing suitable sensing poses for sensor-based algorithms, NBVs, are presented. Further, exploration approaches ranging from naïve random sampling to more sophisticated algorithms are referred. The notion of $\mathcal{C}$-space entropy is introduced. Current knowledge of object inspection is reported.

This thesis focuses on sensor-based exploration using eye-in-hand systems, therefore existing sensors and their configurations are elaborated. Subsequently, the need for developing a new system is demonstrated.

Suitable world model representations are crucial for storing environment changes as well as for safely guiding exploration and inspection. An overview on prevalent approaches for world modeling, i.e. volume-, surface- and point-based approaches, is given.

Furthermore, the field of motion planning is elaborated. Roadmaps, applied as a measure for connectivity, and their construction methods are introduced. Collision detection methods used to compute obstacle-free paths for roadmap construction and path planning are described.

The chapter closes with an identification of initial starting points and current standards that are advanced by the work performed in this thesis.

2.1 Exploration

Whenever information required for task execution is not fully available, an exploration must be performed. The exploration problem usually consists of two components: The map is (partially) unknown beforehand. Therefore, a representation allowing sensor data integration is required. Secondly, measures for guiding (or driving) the exploration are needed, which is closely related to the definition of exploration goals. In exploration, the following issues must be considered:

- map of the environment;
- sensor model and respective map update;
- localization, and
- exploration strategy, i.e. view planning, depending on the task specification.

In this thesis, view planning for $\mathcal{C}$-space, denoted *$\mathcal{C}$-space exploration*, and view planning for tasks in physical space, e.g. object surface acquisition, denoted *$\mathcal{P}$-space exploration* or *inspection*, are differentiated. Commonly, both approaches are summarized as exploration, i.e. increasing knowledge. Additionally, the localization is assumed to be perfect. Nevertheless, the interrelation between localization, mapping, and exploration is close, as most of the information for view planning is derived from previously acquired maps.

In the following, current research on map building, using sensors and corresponding models, is presented. The localization problem in building a map of the environment, i.e. simultaneous localization and mapping (SLAM), is summarized. Furthermore, the exploration strategy and goal definitions for tasks defined in physical space and work space are described.

2.1.1 Mapping

A map is a representation of the environment of the robot. It is used for task execution and can be represented as surfaces, grids, or features. Maps allow robots to efficiently carry out their tasks. Successful robot systems rely on maps for localization, path planning, and task planning. Most exploration methods are designed to build complete maps, while using only parts of them for directing the exploration.

In case the pose x of the robotic system is known, the main goal is the calculation of a map m given sensor measurements z from multiple sources. Mapping does not cover the field of view planning, i.e. the question of where to optimally place the sensor to gain the required knowledge of the surroundings is not addressed.

The criterion for choosing sensing locations is usually an expected information gain. In literature, view planning is also often referred to as exploration. Vice versa, the map itself provides information for planning views, which couples view planning and mapping similarly to the dependency between localization and mapping.

In principle, grid-based maps and feature maps are differentiated. The latter contain task-dependent features, e.g. poses of landmarks. Grid-based maps tesselate the physical space and assign a task-specific value to the cell, e.g. its occupancy state. The textbook of Thrun et al. provides an exhaustive overview on mapping methods [164]. Grid-based maps are used in this thesis, therefore feature-based ones are only briefly addressed.

Mapping is not only relevant in mobile robotics. Even today, most approaches generate 2-D maps for navigation. This 2-D information is used for building 3-D models by a robot. In [55], exploration is used for mapping complete cities.

While early work of Matthies et al. [106] uses stereo and sonar range data to build 2-D grid-based maps, Moravec et al. are the first to publish 3-D grids [113]. In this work, stereoscopic vision is used to update 3-D evidence grids. Borenstein et al. [21] integrate navigation and obstacle avoidance in 2-D occupancy grids for a mobile robot equipped with sonar sensors. While omitting the localization problem, the focus is on real-time obstacle avoidance in a grid representation. A potential is applied for navigation, denoted Virtual Force Field (VFF), attracting the robot towards the goal and repulsing it from detected obstacles, considering the dynamic behavior of a fast mobile robot. A wall-following-method is applied to tackle the local minima, i.e. the robot's being trapped in a dead end. The wall-following-method directs the robot away from this local minima.

The real-time aspect in map building is stressed in [117]. Oriolo et al. use Fuzzy Logic for the efficient building and modification of the environment map of a NOMAD 200 mobile robot, an experimental platform to show the real-time performance of the proposed method in both static and moderately dynamic environments. Murray et al. [114] use stereo for building grid-based maps, applying a very simple update method, which cannot be interpreted physically. The approach is motivated by computational effectiveness, claiming that stereo errors are mostly systematic and not modeled well within a probabilistic framework. Furthermore, the error in the sensor readings mainly depends upon the robot's position and the surface texture, as well as other effects, leading to the following update rule[1]:

$$\begin{aligned} \text{if } i \in OCC(r), \text{ then } G(i) &= G(i) + K \\ \text{if } i \in FREE(r), \text{ then } G(i) &= G(i) - K \end{aligned}$$

[1]This simple rule can be be interpreted as Bayes' Update

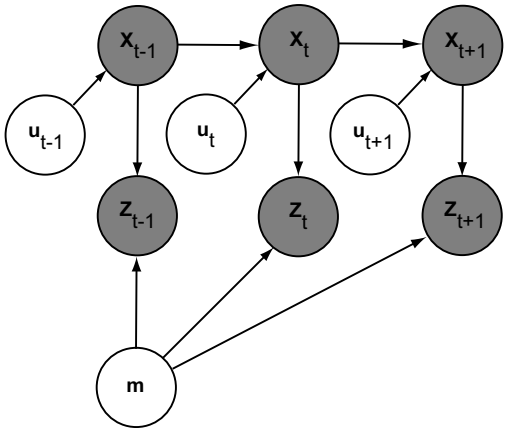


Figure 2.1: In the exploration problem with perfect localization, the shaded variables, i.e. poses x of the robot, as well as measurements z are known at all times. The goal is to estimate the map m and plan the sensing actions u in order to gain as much information as possible in the next step, denoted Next-Best-View (NBV).

where i is an occupancy grid location with value $G(i)$, r is the sensor reading, $OCC(r)$ denotes the region of uncertainty around the sensed obstacle, $FREE(r)$ represents the free region between robot and sensed obstacle, and K is a constant. The values of $G(i)$ are clipped between G_{max} and G_{min} values. The value indicating an unknown grid cell is determined by $(G_{max} - G_{min})/2$.

Hirschmueller et al. [69] use Fuzzy Update for mapping stereo data acquired in random walks, applying a multi-resolution 3-D map, which is uniformly tesselated in x and y at a high resolution, using a lower resolution in z.

Only few surveys on comparing update methods are published. Gambino et al. [56] present a comparison of update models for stereo vision. The authors compare map updates using ultrasonic sensors, applying Probability Theory, Fuzzy Logic, and Fuzzy Measures. Their choice is Fuzzy Logic due to simplistic sensor models and weakening of measurement independency. Cohen et al. [39] describe metrics for finding the appropriate update model.

The most common map representation is an occupancy grid, based on inverse (Bayes' Update) and forward (Belief- and Fuzzy Update) models with beam-based (cf Sec. 2.1.2) sensor characterization. In order to clarify the terms *forward* and *inverse model*, the following definition is given:

Definition 2.1.1 *The environment induces measurements on the sensors, these measurements condition the map. The forward model conditions maps on measurements, whereas the inverse model induces measurements from maps.*

Extensions like Maximum a Posterior (MAP) occupancy grids, which update with forward models (scan-based, see Sec. 2.1.2) are rarely addressed in literature. For details on MAP occupancy grids, the reader is referred to Chap. 9.4 in [164].

The standard occupancy grid mapping algorithm computes the posterior over maps given the data. This method computes the conditional probability

$$p(m|z_{1:t}, \mathbf{x}_{1:t}). \tag{2.1}$$

The map is denoted m. The set of all measurements up to time t is denoted $z_{1:t}$, and $x_{1:t}$ is the path of the robot through the sequence of all poses, i.e. the set of NBVs calculated by the view planning algorithm. The controls $u_{t:1}$ in Fig. 2.1 are estimated using the NBV algorithms. The result from the NBV computation provides knowledge on the maps, which is usually not considered in map update, as the NBVs are calculated before the measurement is done [164, p 285]. In general, occupancy grids are considered probabilistic world models, the map

$$m = \{\mathbf{m}_i\} \tag{2.2}$$

is tesselated into grid cells indexed i. In the following, the sensor models and the corresponding map update are described.

2.1.2 Sensor Models

Sensors are widely used in robotic environments. In mobile robotics, there are contact sensors such as bumpers; for articulated robots these sensors are usually denoted skin sensors. Fernandez et al. [48] apply skin sensors for exploration: the view planning improves because a skin sensor's virtual Field of View is larger than that of sensors in hand-eye configuration. The obvious drawback in this approach is the required physical interaction with the environment. Sensor models are quite complex to estimate and compute. However, using skin sensors is very advantageous in human-robot cooperation in order to reliably stop the robot in the event of unforeseen collision. Internal sensors like accelerometers, gyroscopes, compasses, and inclinometers are used for pose estimation.

Although this thesis assumes ideal poses, these sensors are important for consistent mapping and localization in mobile robotics, a prominent field of application for exploration. The sensors considered in this work are commonly called proximity sensors, as they measure distances to objects without physical contact. Sonar, radar, and laser range-finders based on triangulation, time-of-flight, and phase, fall into this category.

The measurement model of a sensor is categorized as follows:

Beam-based: The sensor's complete FoV in sensing direction is described and computed.

Scan-based: A single point inside the FoV is used for measurement interpretation.

Landmark-based: The location of landmarks is measured and provided as location and orientation in relation to the sensor.

While being closely related to physics, beam-based models are computationally intense due to the ray cast operation required for updating environments. Additionally, these models are not smooth in cluttered environments, e.g. while measuring small objects such as poles, the pose estimation of the beam-based sensor is very inaccurate. Furthermore, statistical independence between individual beams, i.e. sensings, is assumed, requiring care in interpretation and making the computation more complex.

Scan-based models solely consider the end point of a beam. The resulting probability is a mixture of a Gaussian distribution, whose mean is at the distance to the closest obstacle as well as a uniform distribution for random measurements and for maximal range measurements. Scan-based models are highly efficient and smooth w.r.t. to small changes in robot position. While allowing gradient descent and scan matching, these models ignore physical properties of beams.

Landmark-based models are suitable for active beacons (e.g. radio, GPS) or passive markers (e.g. visual, retro-reflective). The standard approach is triangulation, the sensor providing distance, or bearing, or distance and bearing. This type of sensor is usually applied in mobile navigation.

In this thesis, beam-based models are applied. In the following, the application of the sensor models for updating maps is described.

2.1.3 Map Update Rules

Sensor models are applied for updating the maps of the environment. In general, there are multiple methods to perform an update of a map m. For the integration of error-prone sensings into consistent world models, the following three methods can be applied:

1. Bayes' Rule, the first probabilistic approach in mobile robotics using sonar. The work of Elfes and Moravec [47, 107] dates back to the late 1980s.
2. Belief Update, based on Dempster's Rule of Combination, applied to the field of robotics by [143, 123].
3. Fuzzy Fusion [122], applied to stereo fusion in maps [56, 69, 114] for mobile robots with ultrasonic sensors.

In Bayes' Rule, the existence of single hypotheses, each opposing an alternative hypothesis, is mandatory. Based on measurements z, a decision for or against this hypothesis is generated, e.g. the decision between *obstacle* and *not obstacle*.

Dempster-Shafer (DS), i.e. Belief Update, generalizes this approach by allowing ignorance. DS differentiates between sensor information and plausibility. In case a decision cannot be made based on a single measurement but only based on a combination of them (i.e. in complex classification problems), the DS approach is more suitable than Bayes; e.g. for detection and localization of people in robot work cells using sonar, infrared, and capacity proximity sensors [64].

Finally, Fuzzy Update releases the constraint of independent measurements by keeping separate sets for the different membership functions.

For details on the basics of the three theories, the reader is referred to the mathematical annex, Chap. A. In the following, the main implications of the update rules are presented briefly. Bayes' Update is presented more detailed than the other two in order to reflect its significance and widespread application in mobile robotics.

Bayes' Update

In probability theory, Bayes' Rule plays a dominant role in statistics. It relates the conditional probability of $p(x|z)$ to its inverse $p(z|x)$:

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)} \tag{2.3}$$

Bayes' Rule plays a predominant role in probabilistic inference. In robotics, the probability $p(z|x)$ is often called generative model as it describes how state variables $X = x$ can cause sensor measurements $Z = z$. The denominator does not depend on x, therefore this denominator is called Normalizer of Bayes' Rule. In general, the computation can be quite complex and is often neglected. The update is based on the so-called inverse sensor model $p(x, z)$.

The map m usually consists of cells c_i. The measured range value d is the realization of the random variable z; n independent measurements are required to update the map.

$$p(c_i = 1|z = d, x) \propto p(c_i) \cdot \prod_{j \in \{1,..,n\}} p(z_j = d_j|c_i = 1, x_j) \ . \tag{2.4}$$

In general, the independence of these measurements is not given.

The scale factor in the above equations is difficult to calculate, especially in real-time, as it requires consideration of the entire map. In [47], the scale is neglected, which causes inconsistencies in the map, especially if the same object is seen in two consecutive measurements. These inconsistencies have been identified as the major

drawback for the application of Bayes' Update in map building. Therefore, this scale is denominated in odds in [84]. An event w with the probability $p := P(w)$ has the odd

$$O(p) := \frac{p}{1-p} = \frac{P(w)}{P(\bar{w})} \tag{2.5}$$

Further, the log-Odd is defined as

$$L_p = \log(O(p)) \tag{2.6}$$

Log-odds enable a correct computation of the probabilities, as their use evades the computation of the scale.

Alternatively, Belief Functions as provided by Dempster-Shafer can be used. The third possibility is update by Fuzzy Fusion.

Belief Update

Belief Update is based on two ideas: the idea of obtaining degrees of belief for one question from subjective probabilities for a related question, and Dempster's Rule for combining such degrees of belief when they are based on independent items of evidence. The forward model is used for integration of new measurements. A prerequisite for the application of Bayes' Rule is the existence of a complete set of hypotheses as well as a single hypothesis with an opposing hypothesis. If this is the case, Dempster-Shafer and Bayes [41, 139] produce the same result, while their theory and implementation differs. In other cases, Bayes' Rule is not defined. Dempster-Shafer fills this gap as it can process and support uncertain information. In Belief Theory, complete ignorance can be represented, i.e. $m(\Omega) = 1$ (cf. Sec. A.3). One major drawback of the Belief Update is the larger computational effort. Additionally, the theory is not defined for completely contradicting information. More detailed comparisons of Belief and Bayes can be obtained in [44] and [175].

Fuzzy Update

Fuzzy Update applies the forward sensor model. Fuzzy releases the assumption of statistical independence of measurements. Initially, no knowledge of occupancy or free space is available, therefore the respective independent sets are empty. The membership functions for occupancy and free space are updated separately. The update is performed exemplarily for occupancy. The union of the sets ${m^i}_{occ}$, i.e. the current set of occupied cells, and ${m^z}_{occ}$, set of occupied cells induced by a measurement z, is usually performed by applying a t-Conorm (cf. Sec. A.4). Fuzzy Update was used in [56] as the method for integrating stereo data. Hirschmueller [69] applied the same technique for a stereo sensor.

Comparison of Update Methods

The fusion of sensor information to generate task-specific representations is a widespread topic in research. The articles in the proceedings [16] provide a good overview on the current research in information fusion, ranging from robotics to aeronautics and automotive applications. In sensor-based operation, fusion is used to identify *safe-for-motion* areas.

Comparison of fusion results and approaches is difficult. Cohen et al. provide a method for evaluating sensor fusion algorithms based on a quantitative comparison [39]. The authors present a basis for modeling, analyzing, experimenting, and comparing different sensor fusion algorithms based on a ranking basis, enabling selection of the best algorithm for the application.

In this thesis, Bayes', Belief, and Fuzzy Update are evaluated considering their suitability for an identification of *safe-for-motion* areas and for computation of knowledge for directing the exploration.

2.1.4 Localization

In many cases, the robot pose can not be properly measured, especially in mobile robotics, where it is computed based on inaccurate odometry. In general, the pose $x \in \mathcal{R}^6$ contains three translational parameters for the location, as well as three rotational parameters that define the orientation of the robot. The task is to estimate the pose of a robot given the data and the map. Mapping, however, involves a simultaneous estimation of the pose of the robot and the map, i.e. SLAM. SLAM[2] and exploration are key fields of research in the mobile robotics domain. Important approaches are described in the following.

In Fig. 2.2, the SLAM problem is presented. Either grid or feature maps are used. Based on measurements z, the map m and the robot pose x are estimated simultaneously.

The textbook of Thrun et al. [164] extensively describes the localization problem and different solutions to it. Recent SLAM approaches use particle filters [145], their main challenge being the reduction of complexity by minimizing the number of particles. Burgard et al. [29] use position probability grids in order to estimate the absolute position of a mobile robot.

Self-localization with visual sensors is denoted Visual SLAM [34]. This technique is well-suited for localization based on visual and inertial data. Currently, there is a strong need for seamless navigation. In outdoor applications, the Global Positioning System (GPS) is usually used for localization. Indoors, satellite-based systems are not avail-

[2]The website *http://www.openslam.org features a collection of SLAM algorithms. The algorithms are provided as library or matlab code for download*

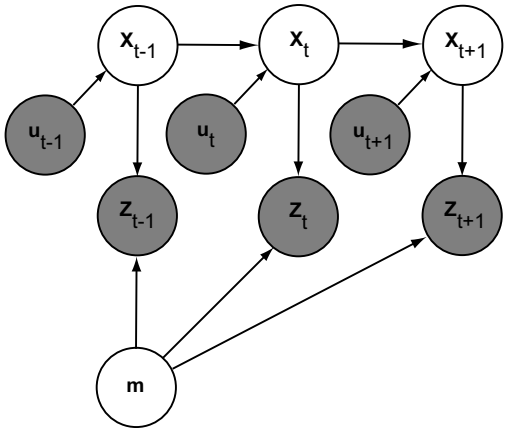


Figure 2.2: In the SLAM problem, map m and robot pose x are unknown and estimated online. In active localization, the controls u_t are unknown. This corresponds to the simultaneous exploration and localization problem.

able. The goal of seamless navigation is to provide methods for localization covering both outdoor and indoor environments. Visual SLAM is a promising approach to that problem, as it does not rely on external localization systems.

In case the robot pose x is measured precisely due to exact robot kinematics or external tracking, perfect localization is assumed. Therefore, the exploration problem can be represented as shown in Fig. 2.1. The task is to compute the control u, which directs the robot to the next pose at which an optimal measurement can be performed.

2.1.5 Exploration Strategies

The computation of the controls u for efficient exploration is subject of this section. In the mobile robot domain, several exploration approaches exist, the following paragraphs naming the most significant ones.

The most naïve strategy is random sampling. A view point is generated randomly and a measurement is performed. The exploration process is denoted as random walks.

Freda et al. [52] present an integrated exploration strategy for mobile robots. This method is based on the randomized incremental generation of a data structure called Sensor-based Random Tree (SRT [118]). Developed in [53], SRT improves the efficiency of the exploration by biasing the randomized generation of configurations towards unexplored areas, which represents a roadmap of the explored area with an associated safe region. Both the information gain and the localization potential are taken into account when evaluating candidate configurations for exploration. In [52, 53], the authors consider noisy sensor

data in constructing the SRT, distinguishing between *safe-for-motion* and *safe-for-planning* areas, biasing the sampling of $\mathcal{C}$ towards unexplored areas. This frontier-based approach utilizes a simple but successful method for defining exploration goals: Regions which have not been updated (i.e. scanned) are chosen for exploration. This coarse estimate of the information gain, called binary gain (Chap.17.4 in [164]), is the most simple exploration algorithm possible, as it always directs the robot to the closest unexplored region.

Sensor-based exploration is usually driven by uncertainty measures (see [185] and more recently [186]). Whaite et al. focus on incomplete data. The goal is an autonomously operating machine that actively senses in order to reduce uncertainty, called autonomous exploration. Whaite et al. [186] deliver two major contributions: the theory for exploration and its implementation. The system is designed entirely bottom-up and does not require any a priori knowledge of the environment. It is the first system to have successfully closed the loop between gaze planning and the inference of complex 3-D models. Superellipsoids are chosen for environment representation, the fitting error is taken as the criterium directing the exploration process.

Thrun et al. present a real-time algorithm [165]. They extend the expectation maximization algorithm to surface models for acquiring compact 3-D maps of indoor environments. A mobile robot equipped with range and imaging sensors, which is later applied in mapping abandoned mines [166], is used as an experimental platform in order to demonstrate the performance of the approach. In both cases, the map is an occupancy grid.

Stachniss et al. [144] use coverage maps rather than grid maps for exploration and mapping, later extended to multiple robots [30]. In [145], the authors apply a decision-theoretic framework that simultaneously considers map and pose of the vehicle in order to evaluate potential actions. Ko et al. [81] perform exploration and mapping with multiple robots and focus on map merging under global uncertainty about the robots relative locations, applying an adapted version of particle filters. Leffler et al. [94] provide a theoretical framework for exploration with a latent structure, analyzing several algorithms and proving matching lower bounds. Mapping and exploration for mobile robots with manipulators is addressed in [153]. The authors target at manipulation of objects in unstructured environments, therefore, the task-specific inspection of objects is detailed.

The next section deals with the computation of the NBV, i.e. determining the optimal sensor pose for performing a measurement.

2.1.6 Computation of Next-Best-Views

While a SLAM algorithm builds a map using the acquired sensory data, an NBV algorithm guides the navigation (exploration) of the robot through poses chosen to provide the best sensory inputs w.r.t. the desired task. Related approaches to compute the views are described in this section.

Tarabanis et al. [160] survey methods for view point selection, motivated by their Machine Vision Planner (MVP) system, which is elaborated in [161]. This system plans model-based views based on known geometry, considering sensor visibility. Occlusion-free view points are adressed in preceding work [162].

In order to choose the proper view planning criterion, three alternating cases must be considered:

- sensing operations are expensive, whereas motion is cheap;
- motion is expensive, sensing is considered a cheap operation;
- sensing and motion are both expensive to perform.

The first case relates to the *art gallery problem*, well-known in computing science and described later in this section. While the original problem assumes an infinite FoV and panoramic view (360° vision), these assumptions never apply in reality.

The second case, expensive travel, relates to the *traveling salesman problem*. Here, sensing costs are negligible.

The third case, both operations being expensive, is the most realistic assumption. A vivid example is a mobile robot acquiring scans of large environments, e.g. buildings or streets: Typically, a single scan takes up to an hour, whereas the performance of the mobile robot is limited by its on-board power, i.e. motion cost. The task is to choose optimal view points, i.e. scanning locations, and thus minimize their number and the distance between them. This problem is known as the *watchman route problem*. Recent work of Wang et al. [181, 182] addresses this problem in a theoretic way from the complexity point of view.

Blaer et al. [17, 18] show extensions to early work from the AVENUE project for urban modeling [8] as well as modeling of historic sites [7]. The authors [18] apply a two-stage approach: generation of view points from aerial data, followed by a refinement of these view points. The first step applies a voxel-based improvement of the surface-based Positional Space Approach of Pito [125]. Pito extends early work at the GRASP laboratory by [108], however, not considering motion cost. Additionally, a turn-table approach is used, making sensing range considerations unnecessary as the sensor's vision is larger than the turntable diameter. Frueh et al. [55] use airborne data in combination with ground data to build city models.

In 3-D modeling, the key challenge is to completely acquire the surface of an object, using only a minimum number of sensor poses by computing an optimized set of these poses. The so-called Set-Covering Problem (SCP) is an open issue for the View Planning Problem (VPP). The SCP is NP-complete[3] for decision, NP-hard for approximation; NP stands for Non-deterministic Polynomial time and describes the complexity class of the problem. For details, the reader is referred to the survey of Scott et al. [136].

As fundamental insights can be gained for the VPP in general, the basic problem is described in more detail: The *art gallery problem* was introduced by Victor Klee in 1973 in a discussion with Vasek Chvatal. An art gallery is modeled as a polygonal region in the plane. A guard (camera position) in the gallery corresponds to a point in the polygon. Given a guard with 360 degrees vision, the problem is to determine the minimum number of guards required to cover the gallery. In [119], the art gallery problem is discussed in detail. This book explores generalizations and specializations in these areas, covering theorems on orthogonal polygons, polygons with holes, exterior visibility, visibility graphs, and visibility in three dimensions.

Finding the minimal cover of any specific polygon is NP-hard. There is a well-known approach to finding a guard cover that guarantees no more than $n/3$ guards, called art gallery theorem: *For a simple polygon with n vertices, $\lfloor n/3 \rfloor$ cameras are occasionally necessary and always sufficient to have every point in the polygon visible from at least one of the cameras.* The theorem also assumes that the guards have an unlimited distance of vision and that they can view a wall at any grazing angle. None of these assumptions are true for most sensor systems. Gonzalez et al. [59] propose a randomized method for approximating solutions to art gallery problems. In [18], this approach is extended to 3-D by using the grazing angle, as already presented in [60] for 2-D maps.

In case of mobile robots, overlapping views should be considered in order to decrease model error induced by uncertain localization. In [135] this registration constraint is elaborated. In general, local registration techniques like Iterative Closest Point (ICP) suffice, the reader is

[3] In complexity theory, the NP-complete problems are the most difficult problems in NP ("non-deterministic polynomial time") in the sense that they are the smallest subclass of NP that could conceivably remain outside of P, the class of deterministic polynomial-time problems. The reason is that a deterministic, polynomial-time solution to any NP-complete problem would also be a solution to every other problem in NP. The complexity class consisting of all NP-complete problems is sometimes referred to as NP-C. A more formal definition is given below. One example of an NP-complete problem is the subset sum problem which is: given a finite set of integers, determine whether any non-empty subset of them sums to zero. A supposed answer is very easy to verify for correctness, but no one knows a significantly faster way to solve the problem than to try every single possible subset, which is very slow. Source: http://en.wikipedia.org/wiki/NP-complete 2007-08-15

referred to general registration techniques described in [28].

Planning views on known objects for measurements or visualization is a common topic in computer vision. Most approaches are based upon polygonal data of objects [22] and extend the approach by including a registration component [14], following geometric reasoning. For view planning for object recognition Arbel et al. apply an entropy map in order to reduce ambiguities in recognition of known objects [10].

2.1.7 Exploration of Work Space: $\mathcal{C}$-space Entropy

Yu et al. [190] introduce the notion of $\mathcal{C}$-space entropy (cf. Sec. A.1 for details on the measure of entropy). For sensor-based motion planning, both the physical space $\mathcal{P}$ and the configuration space $\mathcal{C}$ are considered a stochastic process. The assumption of an obstacle distribution in $\mathcal{P}$ or in the stochastic geometry model induces a probability distribution of possible $\mathcal{C}$-space instances. Hence, $\mathcal{C}$-space becomes a stochastic process. The knowledge of this process is captured by the notion of Shannon's entropy [140], called $\mathcal{C}$-space entropy.

A sensing action to acquire the most additional knowledge of $\mathcal{C}$-space, i.e. to maximally reduce $\mathcal{C}$-space entropy, called Maximum Entropy Reduction (MER), is chosen based on the view planning.

The robot is denoted $\mathcal{A}$. The physical space is stated $\mathcal{P}$, the unknown parts of this space are named $\mathcal{P}_{unk}$. A robot configuration, or a point in $\mathcal{C}$, is denoted by $\mathbf{q}$.

The kinematics and the geometry of the robot induce a probability distribution for the $\mathcal{C}$-space via the mapping function $\mathcal{A}(\mathbf{q})$. The $\mathcal{C}$-space entropy $H(\mathcal{C})$ is given by:

$$H(\mathcal{C}) = -\sum_{Q_1=occ,free} \ldots \sum_{Q_n=occ,free} P[Q_1,...,Q_n] \log P[Q_1,...,Q_n] \qquad (2.7)$$

$P[\ ]$ denotes the probability of the random variable Q_i of a configuration q_i being free (=0) or in collision (=1), while n is the number of sampled robot configurations in $\mathcal{C}$-space. The expected entropy reduction $\triangle H(\mathcal{C})$ after sensing a region $\mathcal{P}$ (occupied/free) can be computed. In [191] the Poisson Point Model [149] is used as stochastic geometrical model for the physical space because of two reasons:

- more complex models are too complicated in $\mathcal{C}$-space computation;
- $p(\mathcal{A}_{unk}) = e^{-\lambda \cdot \mathcal{A}_{unk}(q)}$ is intuitive. The larger the volume of the robot in unknown physical space, the less likely it is that this volume will be free.

The Poisson Point Model can be characterized by points uniformly distributed in space. These points are obstacles in the sense of robot

motion planning, so the probability of an arbitrary set $\mathcal{B} \in \mathcal{P}$ being free of obstacles can be described by:

$$p(\mathcal{B}) = P[\mathcal{B} \subseteq \mathcal{P}_{free}] = e^{-\lambda \cdot vol(\mathcal{B})} \ .$$

The variable λ is the intensity of the Poisson Point Model in the value range $\lambda \in]0, \infty[$. In the spatial case, it can be interpreted as a density. In order to produce decision rules for view planning, the expected information gain (IG) it is formulated considering all possible sensings s:

$$IG_{\mathcal{C}}(s) = -E\{\underset{s}{\triangle} H(\mathcal{C})\} \ .$$

$H(\mathcal{C})$ denotes the current $\mathcal{C}$-space entropy.

The computation of the expected IG for different view planning methods is described in [176, 177, 188]. The deepest insight in $\mathcal{C}$-space entropy can be extracted from [192]. Especially in [176, 177], the authors detail the calculation of the entropy for a point FoV, a beam sensor, and a generalized non-zero FoV sensor, which is closest to the real life case. The latter two take occlusion into account, which makes the computation more expensive but improves the exploration results. These algorithms are, without exception, based on the simplistic Poisson Point Model and ideal sensing, i.e. except for the limitation of the sensor's FoV, no sensor characteristics are considered.

The MER criterion [192] results in much more efficient $\mathcal{C}$-space exploration performance than $\mathcal{P}$-space based view planning criteria, maximizing the unknown physical volume in each view [86]. From a $\mathcal{C}$-space perspective, the MER criterion consists of two aspects: sensing actions are evaluated in $\mathcal{C}$-space (geometric aspect), and their effects are evaluated in an information theoretical sense (stochastic aspect). Wang et al. [180] investigate to which extend the exploration performance is attributable to the geometric component or the stochastic aspect of MER. While a major part is attributable to the pure geometric aspect, the stochastic aspect, despite being based on simple assumptions, results in a moderately more efficient $\mathcal{C}$-space exploration.

2.1.8 Exploration for Modeling: Inspection

In this thesis, *inspection* denotes the exploration process for modeling objects in $\mathcal{P}$-space. Laser scanning range sensors are widely used for high-precision, high-density 3-D reconstruction and inspection of the surfaces of physical objects. Automatic modeling involves planning a set of views, physically altering the relative object-sensor pose, taking scans, registering the acquired geometric data in a common coordinate frame of reference, and finally integrating range images into a non-redundant model.

The modeling problem has been a major field of research in robotics in the mid-nineties [125, 126, 160, 161, 163]. Early work dealt with

automatic modeling using turn-tables, which limit the search space significantly. Although reducing the complexity of the problem, the sensor motion limitations cause problems for non-convex objects.

Many works try to limit the view space, i.e. the number of potential views. Liska et. al [103] apply an entropy-based view point selection measure.

Additionally, the computation of visibility is a measure for directing the inspection task. The works [111, 152, 194] use surface-based models for computing visibility. Extensions to mobile robot model acquisition are addressed in [62]. The authors approximate environments using flat surfaces acquired with laser-range scanners on a mobile base, incorporating a scan-matching method in order to account for pose uncertainty. Meshes are generated and simplified to reduce the complexity of the resulting model. For view planning with complex robots, volumetric methods as published in Schauffler et al. [133] seem more promising.

Approaches to partial automation of the scan-register-integrate tasks exist, while computation of the task-specific NBV remains an open problem. Scott et al. [136] survey and compare view planning techniques for automated 3-D object reconstruction and inspection by means of active, triangulation-based range sensors. The termination criteria of an inspection are discussed and identified as open problems. Especially surface-based methods lack a sufficient termination measure, because without model knowledge, real holes cannot be differentiated from insufficient data acquisition.

Coverage of the object is another open problem. Exact cellular decompositions represent a robot's free space by dividing it into regions with a simple structure; the sum of the regions fills the free space. These decompositions have been widely used for path planning between two points, but can also be applied for mapping and coverage of free spaces. Acar et al. [4] define exact cellular decompositions in which critical points of Morse Functions indicate the location of cell boundaries, approaching the object coverage problem.

Banta et al. [12] focus on design and implementation of a system capable of automatically reconstructing a prototype 3-D model from a minimum number of range images of an object. This model-based approach iteratively renders range and intensity images of the model from a specified position, assimilates the range information into a prototype model, and determines the sensor pose from which an optimal amount of previously unrecorded information may be acquired. The termination criterion of the object inspection method is given by model accuracy: When the model achieves a certain resolution, the acquisition is terminated. The authors test successfully on several synthetic data models, with each set of results being reasonably consistent with an intuitive human search.

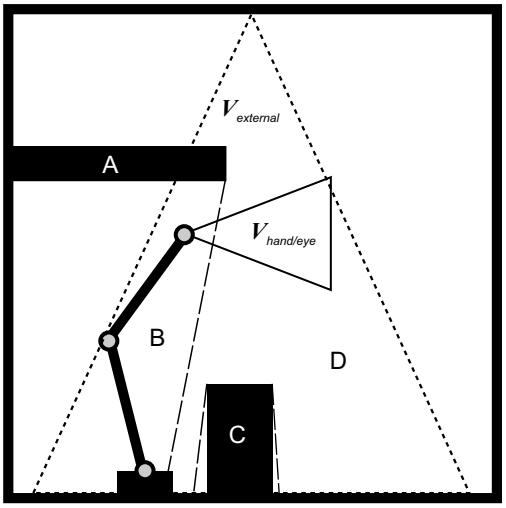


Figure 2.3: A two link robot in physical space. A sensor with field of view $\mathcal{V}_{hand-eye}$ is mounted at the end-effector, another sensor $\mathcal{V}_{external}$, placed fixed above the robot, surveys the scene. The obstacles A and C occlude parts of the scene.

Several methods of reconstructing 3-D scenes from range images have been proposed. The factor limiting a fully automated reconstruction process is the lack of a sufficiently scalable planning algorithm for the acquisition step. Klein et al. [80] present a technique of iteratively planning the next view for a range scanner in an initially unknown large indoor environment. The authors propose an objective function based on the analysis of occlusions which takes into account both a quality criterion and the cost of the next acquisition. Additionally, the parameters of the next view have to be computed efficiently for a large search space with eight DoFs.

2.2 Sensor Systems and Configurations

In the field of robotics, perception is required in various tasks such as object modeling and recognition, visual servoing, exploration, collision avoidance, path planning, simultaneous localization and mapping, and many more. Referring to an overview on optical perception systems, Besls [15] survey gives a highly detailed comparison of radar, triangulation, and other optical sensing principles, still valid today, and already stating the emerging need for contact-less 3-D perception, especially in robotics.

2.2.1 Sensor Configuration

In general, two sensor placements, i.e. configurations, are differentiated: *hand-eye* and *external*, depicted in Fig. 2.3. In the following, the

two configurations as well as the benefits of integrating them are discussed. Furthermore, sensor systems are described, focusing on range measurements.

Hand-Eye Configuration

When a sensor is mounted on a robot, this configuration is denoted hand-eye. It is preferred for visual servoing. The task-specific resolution and the visibility of the target area is advantageous for modeling. The sensor's gaze direction moves with the robot, therefore pose precision and calibration error must be considered. Another drawback is the limited FoV; in Fig. 2.3, the obstacles A and C are currently not visible to the robot. Therefore, view planning is required and crucial for a successful and efficient task execution.

External Configuration

In this configuration, a sensor is mounted fixed within the work space. It is crucial to choose the right sensor placement in order to avoid occlusion. In Fig. 2.3, region B is occluded by obstacle A, limiting the external sensor's vision on the work cell.

Combination of Configurations

For optimal work cell surveillance, a combination of both configurations is required. The hand-eye configuration is optimal for modeling objects and grasp planning, as it keeps the object of interest in focus. The drawback of its limited vision is compensated by an external sensor. This monitors regions which are not visible to the robot hand. The moving robot must be detected and subtracted from the acquired data when the cell is modeled. Although this approach is not described in detail in this thesis, the test-bed (cf. Chap. 8 and Fig. 2.4) is equipped with external sensors, in this case an optical tracking system, for external surveillance.

2.2.2 Sensor Pose

In general, it is not possible to acquire a 3-D model of the environment with one single measurement step (be it a laser scan or visual information acquisition) due to self-occlusion and/or object size. The 3-D geometrical information gathered from a particular vantage point, limited by self-occlusion of the object, is actually called 2.5-D information. Hence, multiple 2.5-D views must be acquired in order to subsequently merge them into a single 3-D model. Existing systems are primarily differentiated by the way this merging process is accomplished: data

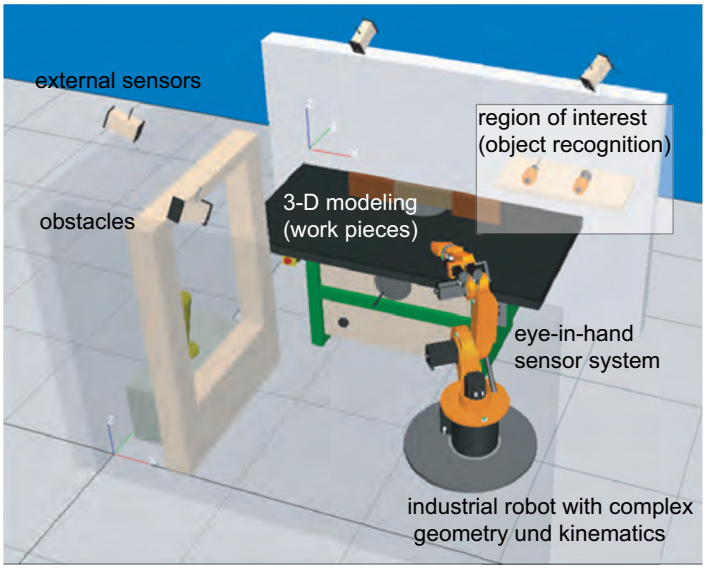


Figure 2.4: Typical SME work cell with external sensors and sensors in hand-eye configuration.

can be acquired from unknown positions, the different 2.5-D views being registered based on overlapping domains (usually 20% of a view). Alternatively, the position and orientation (i.e. pose) of the sensor is measured while sensing. There is no need for strongly overlapping views and intensive computing; the 3-D data sets (point clouds) are readily applicable for online visualization and real-time processing.

2.2.3 Sensor Systems

Numerous 3-D digitization systems work with turn-tables, e.g. Reed et al. [126]. A sensor device is precisely rotated around the object (or vice versa) in order to merge 2.5-D data sets into a 3-D model using the rotational angle as a reference. However, the size of the objects to be scanned is physically limited. It is more convenient and versatile to 3-D digitize an object by simply sweeping the sensor over the object's surface without limitations. In this case, the object can be more complex, and the user solves the problem of planning the views. The sensor must be light-weight, as it is to be hand-guided by the user.

Most systems use only one type of sensor assigned to a specified use, e.g. a hand-guided system using a laser-stripe profiler sensor (LSP) attached to a passive manipulator for pose estimation is described in [49, 67].

Today's commercial vision system manufacturers offer a diverse range of relevant solutions, partly depicted in Fig. 2.5:

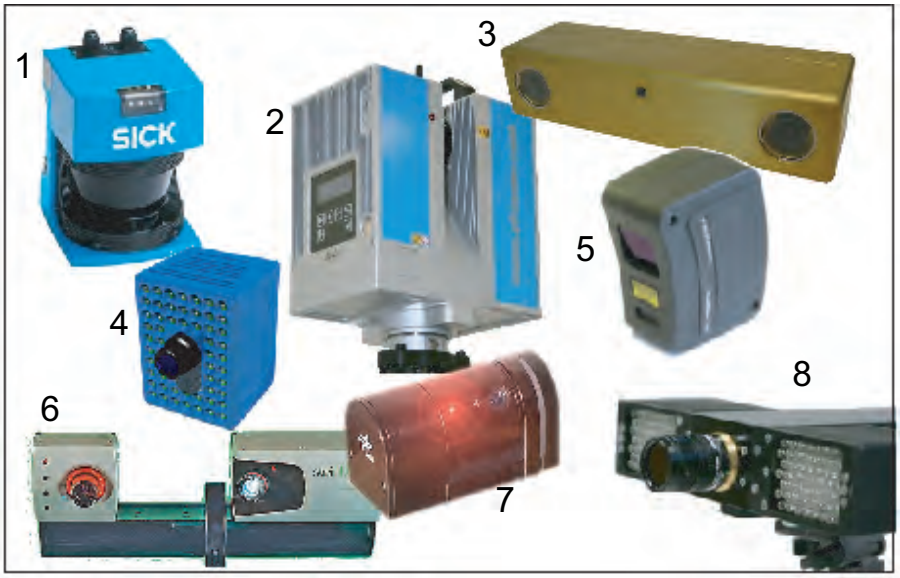


Figure 2.5: Diversity of range sensors: SICK LMS 200 (1), Z+F Imager 5006 [89, 54] (2), Point Grey Bumblebee 2 (3), SwissRanger-3000 (4), 3D Scanners ModelMaker Z (5), Steinbichler Comet IV (6), DLR-LRS (7), PMDTec PMD[vision] 19k (8).

1. SICK[4] provides 2-D laser scanners. The LMS 200 (1) is the standard range sensor used in mobile robotics today;
2. Z+F[5] produce the Imager 5006 (2), a system for sensing outdoor and indoor scenes with measurement distances of up to 80 m;
3. Point Grey Research Inc.[6] focus on single cameras and stereo vision systems (3);
4. CSEM[7] manufacture the SwissRanger-3000 (4), a 2.5-D range sensor.
5. 3D Scanners Ltd.[8] developed a laser-stripe profiler mounted on a FAROarm; This system is used for manual inspection and quality control. It applies an optical filter to increase the performance;
6. T-Scan (6), a product of Steinbichler Optotechnik GmbH[9], is a highly accurate optically tracked scanner for quality control;
7. DLR Laser-Range Scanner (7) [61];

[4]http://www.sick.com Link: 2007-05-17
[5]http://www.zofre.de Link: 2007-05-17
[6]http://www.ptgrey.com Link: 2007-05-17
[7]http://www.csem.ch Link: 2007-05-17
[8]http://www.3dscanners.com Link: 2007-05-17
[9]http://www.steinbichler.de/en Link: 2007-05-17

8. PMD Technologies[10] produce sensors based on the Photonic Mixer Device (PMD) principle (8). Their principal of operation is similar to the CSEM SwissRanger;

9. Isra Vision AG[11] provides numerous industrial multi-line triangulation sensors;

10. Polhemus[12] developed Fastscan and Fastrak, hand-held laser stripe profilers referenced by an electromagnetic motion tracker;

Furthermore, highly specialized vision systems [87, 131] can be found in research, their specialization limiting the field of applications for each system. Gockel et. al [58] develop a structured light sensor suitable for eye-in-hand configuration.

All systems above are based on one sensing principle, be it stereo, laser, or touch. The specifications of these systems are limited and individually tailored to the desired application: there are laser-range scanners for mobile robots, sonar proximity sensors for obstacle avoidance, cameras for pose or texture estimation, and stereo cameras for 2.5-D sensing. Neither do they combine the strengths of the individual principles nor do they evade weaknesses by combination.

Multisensory features, enabling a robot to perform exploration, object recognition or surface feature detection, and tracking simultaneously, do not yet exist in robotics.

Such systems are complex to design. In robotics, the main criteria are accuracy, small size, low weight, and high frame rates. Such a development requires knowledge of mechanics, electronics, and software engineering, all yielding to mechatronic design principles [65].

In the industrial robotics domain, there is a strong need for perception systems that are applicable in flexible work cells. The variability in application requires adaptability in sensing, because sensors are the only source of information during task application. The following procedures must be performed:

- object recognition and pose estimation;
- object inspection, i.e. estimation of geometry;
- exploration, i.e. traversing of and acquiring knowledge on (partially) unknown environments.

In the next paragraph, the previous work on sensor systems and concepts is presented.

[10] http://www.pmdtec.com Link: 2007-05-17
[11] http://www.isravision.com Link: 2007-05-17
[12] http://www.polhemus.com Link: 2007-05-17

Previous Work at DLR At DLR, the development of the laser-range scanner dates back to the year 1986, when the principle was first patented. The intention was to develop a sensor which can be integrated into the robot end-effector, equipped with a gripper, in order to automatically grasp objects approaching the gripper. A single point laser was integrated into the multisensory gripper that was operated in tele-presence mode during the German space mission D2 in 1993. The task was to grasp a free-floating object using the first robot operated in space. Local autonomy[13] was necessary to compensate the long transfer times of commands from ground station to robot. Later, this point sensor was enhanced by a motor, enabling 2-D vision. This completely integrated system with a short sensing range has proven to be robust and reliable (cf. Fig. 2.5 (7)).

A combination of this successful system with longer range sensors and larger FoVs was aspired. The first envisaged application for such a multisensory system was 3-D modeling, which induced a system integrating the most common vision principles for range acquisition: laser range sensing, laser-stripe profiling and stereo vision.

In a first step, 3-D modeling of small objects was performed, i.e. experiences with a hand-operated system were gained before developing a robotic sensor. This system is described in Chap. C as it is the basis for the robotic version of the system presented in Sec. 7.2. This specifically designed multisensory system is developed based on specifications derived from the exploration, modeling, and object recognition tasks.

2.3 Model Representations

Model building is a fundamental task in robotics in general. Knowledge of the robot's environment is required for motion planning as well as task planning. In case of sensor-based operation, the focus is set on incremental methods, as the model of the environment changes during operation and more information becomes available, requiring changes in task and motion plans. The basic problem is to integrate data (range and robot location) acquired in sensing operations performed by robots equipped with sensors in their (partially) unknown cluttered environments into a precise and reliable representation of this environment. In reality, this is a difficult problem due to data amount[14], data noisiness, and environment properties such as non-ideal surfaces.

In this section, the related work in this field, yielding the implementa-

[13]The concept of local autonomy developed at DLR assigns decisions on actions based on direct sensor feedback, while the global control of the mission is guided by the operator. This concept is originally designed for space applications, however, it is commonly applied in medical robotics today.

[14]The environment representation accumulates data over time.

tion described in Chap. 5, is summarized. The textbook [105] provides a good overview on 3-D model generation in robotics and computer vision. In this thesis, the approach is limited to calibrated sensors at known poses.

In the following, the different abstraction levels for world models are given. First, point models are described. Further, surface reconstruction methods are addressed. Finally, volumetric model building is elaborated.

2.3.1 Point Models

Point clouds are the most simple representations of accumulated information. In general, point clouds can be represented in any dimensional space; this thesis focuses on 3-D environments. Drawbacks are the large amount and inhomogeneous spatial arrangement of the data due to non-constant movements when mounting the sensor on articulated platforms. Nevertheless, point representations can be very efficient for fast rendering of large data sets, so-called splatting. The work of Rusinkiewicz and Levoy [129] is a good example for such an implementation[15].

While point clouds are a set of (unorganized) points $X = x_1, ...x_n$ in 3-D, motion planning requires the computation of collision-free paths. Knowledge of the borderline between free space and occupied space is required. This information cannot be obtained from point models without computing an outline or hull, enclosing a cluster of points that form one object. Therefore, point models are rarely used in motion planning as the advantages they offer for visualization do not apply when closed entities are preferred.

2.3.2 Surface Models

Regarding the abstraction level, surface models are one level above point clouds. The task in surface reconstruction is to find a surface M' approximating M, based on a set of (unorganized) points close to or on the surface M. Four techniques address this problem:

- parametric reconstruction methods;
- implicit reconstruction methods;
- hybrid implicit and parametric methods;
- function reconstruction methods (rare).

[15]The modeling process is not dynamic in this particular work, as the computation of normals is required for visualization. Therefore, the complete data set is known before processing it into the world model.

The most commonly used parametric surface approximation is the triangle mesh. Early work from Hoppe et al. [75] deals with the problem of generating meshes from unorganized points, which usually occurs when scanning objects hand-guidedly. In Bodenmueller et al. [19], this problem is addressed for the DLR 3D-Modeller system (cf. Chap. 7). The digital Michelangelo Project[16] by Levoy et al. [95] is another example for successful parametric surface model generation.

Polygonal surface models, especially triangle meshes, are easy to compute and render. Nevertheless, polygonal data sets are prone to errors like holes [42], gaps, and self-intersections, requiring significant post-processing of the data. Falsely detected holes in the model make collision-free path planning for robots impossible, thus requiring hole filling methods [97]. Such data sets are denoted polygonal soups, as they offer no warranty for their structure, e.g. connectivity and segmentation. The missing structure, apparent for many surface models especially when based on sensor data, is a drawback for the use of these models for planning.

An approach to solve this problem is the use of implicit surfaces, which can be generated from polygonal surface descriptions. Implicit surfaces approximate a given geometry, depending on the feature size to consider. The work by Shen et al. [141] on implicit surfaces is based on [169]. The general constraint is to find a function $f(\mathbf{x})$ whose zero set approximates the objects' surface, given N object points at positions $\mathbf{p}_i, i \in [1, N]$. The major drawback of implicit surfaces is the non-recursive approach. The integration of new points into the model, relevant in sensor-based model acquisition, results in recalculation of the entire 3-D model. More details on constraints induced by approximation of polygons and contours are provided in [141].

2.3.3 Volumetric Models

Samet [130] provides an extensive overview on spatial data structures. Although this book was published almost 20 years ago, it contains the basic ideas on structures and methods for indexing, mostly in a GIS[17] context. The primitives for representing volumes can be arbitrary. Mostly, researchers apply cubes or spheres as primitives.

Volumetric models are represented in uniform or hierarchical grids. A member of the latter class is an octree (cf. Fig. 5.13 in Chap. 5.5.3). In octrees, each node has 8 sub-nodes, whereas the root contains the full 3-D space. This multi-resolution representation is suitable for storing

[16] The Digital Michelangelo Project was a two year project from 1997 until 1999. The goal of this project was 3-D scanning of sculptures and architectures of Michelangelo. 10 statues, including David, were scanned and processed. More information can be obtained from http://graphics.stanford.edu/projects/mich/ (Link: 2007-02-17)

[17] GIS: Geo-Information System, providing 2-D maps with semantic information.

large data sets at acceptable resolution. A uniform grid model fills the object with voxels of the same volume.

Bradshaw uses sphere trees for modeling objects [23, 24, 25]. This work, however, does not consider an incremental generation of the tree, which would be required in sensor-based applications.

Volumetric models can be generated in multiple ways, which differ by their level of abstraction. On the lowest level, surfaces are represented as volumes by means of voxelization of the surface boundary, a so-called surface voxel model. The visual hull of an object (O) is defined as a complete voxel model, which can be constructed from extruded 2-D to 2.5-D silhouettes. Initially, a bounding box (BB), e.g.

$$dim_{(x,y,z)}(BB) > max(dim_{(x,y,z)}(O)), \tag{2.8}$$

with dimensions ($dim_{(x,y,z)}$) (larger than the object to be modeled) is completely filled with voxels. Then, the voxels are extruded, performing a so-called volume carving.

In case of axis-aligned cubes, there is no difference in complexity between cubes and spheres. However, spheres are better suited, as they do not require an orientation to be considered when computing distances for collision detection.

2.4 Motion Planning

The general motion planning problem is summarized in Tab. 2.1. In the model-based case, the robot environment is completely known, therefore no sensors are needed for planning. Exploration is not required. In the sensor-based case, the robot must acquire its environment using sensors. Motion planning is usually performed in configuration space, which is therefore often denoted *planning space* as opposed to *physical space*, i.e. the cartesian space.

A common way of planning motions is the implementation of roadmaps.

When incrementally building roadmaps of the planning space based on sensory data, the roadmap construction methods simultaneously direct the exploration of unknown areas. This means that the success of expanding the roadmap is a measure for the progress of the exploration. Therefore, the choice of roadmap construction method and representation has a direct impact on performance and quality of the exploration.

2.4.1 Roadmaps

The configuration space usually has a high dimensionality, based on the number of DoFs, therefore the most common approach to storing

Table 2.1: The general motion planning problem

Motion Planning	Model-based	Sensor-based
Environment knowledge	complete	partial
Robot perception	none	sensors

$\mathcal{C}$-space information is the use of roadmaps, i.e. graphs, in which the robot's motion is planned.

Overview Sampling-based algorithms have been widely used in solving motion planning problems. Randomized potential field methods [13] perform very well in relatively uncluttered $\mathcal{C}$-spaces. The failure of potential field methods in simple situations, however, is a drawback of the method. Probabilistic methods such as the Probabilistic Roadmap Methods (PRMs) perform well in a number of situations, e.g., [78] and [189], but the sampling technique has high influence on the mapping result [76], requiring biased sampling. Furthermore, tree-based planners are randomized path planning methods, incrementally constructing trees that explore a connected region of $\mathcal{C}$-space. A tree-based path planner, denoted Rapidly-exploring Random Tree (RRT), was developed by LaValle and Kuffner in [88] and [92]. By only exploring the relevant portion of configuration space needed for the query, the Expansive Space Tree (EST) [77] works well for single-query problems.

Roadmaps break down the complexity of planning space quite efficiently, reducing the motion planning to a graph search problem. The representation of planning space in roadmaps requires construction methods. The general goal is to obtain a roadmap representation which is as sparse and complete as possible w.r.t. the desired task. Due to the high dimensionality of the problem, sampling methods to build the graph are preferred, ranging from naïve to biased, task-specific approaches.

Although it is generally computationally impractical to develop a complete knowledge of a configuration space, the selection of samples, each providing information about the configuration space, is crucial. Burns and Brock [32] present an approach to finding optimal samples by considering an approximate configuration space model. This selects samples that are maximally relevant to the planning task by applying a utility function defined by entropy-based information gain. Exploration is not addressed, as the environment is assumed to be completely known, eliminating a direct application in sensor-based planning and exploration. Especially the utility function needs adjustment in case such an application is envisaged.

Hierarchical Construction Methods The Hierarchically Generalized Voronoi Graph (HGVG) is a roadmap developed for sensor-based exploration in unknown environments [37]. As the HGVG is one-dimensional, the motion planning problem is also one-dimensional for any kind of dimensionality of the robot. The mathematical foundations of the method are described in [37]; the incremental construction method is used for exploring the environment in [38]. The authors state that the method can be used as a basis for sensor-based planning algorithms, as the incremental construction procedure of the HGVG requires merely local distance sensor measurements. They show simulations and experiments on a mobile robot equipped with ultrasonic sensors to verify this approach. Yeong et al. [93] extend the HGVG approach to convex-shaped bodies. Another promising method for mobile robots is described in Oriolo et al. [118]. Their method first uses PRMs to sample random configurations as milestones and then grows trees from each node, combining PRMs and sampling-based tree methods, called Sensor-based Random Tree (SRT).

Probabilistic Construction Methods Sampling-based motion planning approaches reveal the implicit connectivity of a configuration space by selecting and connecting sets of configurations. Nissoux et al. [115] introduce visibility-based roadmaps. The configuration space is sampled and based on a visibility constraint, the nodes are inserted into the roadmap. The roadmap contains guard and connection nodes. Guard configurations cover a region of $\mathcal{C}$ containing all configurations connectable to the guard by a straight line. Only configurations outside the cover of the guard or connecting two guards are inserted into the roadmap. The high computational cost for construction is justified by significantly smaller roadmaps.

Hsu et al. revisit the basics on PRM in [76], trying to establish the probabilistic foundations of PRM planning and re-examining previous work in this context. The success of PRM planning depends on the assumption that the $\mathcal{C}$-space of a robot often inherits visibility properties that do not directly depend on its dimensionality. In case of high dimensional $\mathcal{C}$-spaces in environments represented by a large amount of triangles, the high cost of computing an exact representation of the free space, which is the collision-free subset of $\mathcal{C}$, forbids the use of algebraic planners. PRMs are an extremely simplified representation of the free space. Nodes in PRMs are sampled configurations based on a probability measure. The edges are usually simple collision-free paths, i.e. straight lines, between the nodes. PRMs are suited very well for exploration problems, as sampling can be performed iteratively. The performance is critically influenced by the sampling methods, in most cases biased sampling is most favorable or even required.

Rimon et al. [128] present an incremental navigation algorithm for gen-

eralized articulated robots. They state that the robot may start with no a priori information about its environment, and is guaranteed to find the goal if it is reachable, or to halt otherwise. The algorithm constructs a roadmap based on distance data collected online and encoded as a repulsive potential field. The incremental behavior is achieved using two abstract sensors: a critical point detector and a minimum passage detector. The authors show which environmental features must be measured by the detectors for a planar-body robot. The close relation between task, i.e. exploration, and task representation, i.e. roadmap based on potential field method, is clearly visible. However, the work lacks experiments in real environments.

2.4.2 Progress of Exploration

A small number of measures for monitoring the progress of exploration methods, i.e. roadmap expansion, exists. These can be divided into local and global methods. In most cases, naïve random sampling is the most efficient method: Random samples in $\mathcal{C}$-space are drawn and checked for collision with known and unknown areas, which results in a global estimate of $\mathcal{C}$-space knowledge. Morales et al. [112] give local and global metrics based on roadmap expansion attempts (locally and globally) by computing the volume covered by roadmaps. This method achieves better estimates for $\mathcal{C}$-coverage than naïve sampling and has been applied to RRT and other methods. The authors show results in ideal environments commonly used in motion planning, yet lack remarks on applications on real systems.

2.4.3 Collision Detection Methods

Collision-free path planning requires efficient methods to determine whether objects occupy the same volume at the same time. Additionally, information about objects approaching each other is useful in dynamic environments. There are several algorithms for collision inference. They usually apply a broad phase, where possibly colliding objects are identified, and an inference phase, in which collision points or penetration depths are computed. In the following, the abbreviations for Oriented Bounding Box (OBB) and Axis-Aligned Bounding Box (AABB) are used to describe the bounding volume. Lin and Gottschalk's survey [98] describes methods such as Lin-Canny closest features algorithm, V-Clip, I-Collide, OBB-tree, and Kinetic Data Structure (KDS)[18]. A detailed depiction of closest feature, I-Collide, and OBB-tree including applications is the scope of [99].

[18] An overview on collision detection libraries and scientific papers can be obtained from the GAMMA (Geometric Algorithms for Modeling, Motion, and Animation) group at Department of Computer Science, University of North Carolina, http://www.cs.unc.edu/ geom/collide/

The Lin-Canny closest features algorithm [101, 102] keeps the pair of closest features (vertices, edges, or faces) between two convex polyhedra moving through space. It makes use of the fact that the current closest features are near the previous closest features, achieving a near-constant query time. The distance between two polyhedra is defined once the closest features are known, a collision is detected if the distance is shorter than $\epsilon > 0$.

Baltzakis et al. [11] present a method which infers scene structure information from sensory data. The proposed methodology is applied to robot motion planning and collision avoidance tasks by using a suitably modified version of the vector field histogram algorithm.

The SOLID library, provided with the textbook [171] applies an extension to the Gilbert-Johnson-Keerthi (GJK) [57] inference algorithm, denoted Incremental Separating-Axis (ISA) GJK [170]. SOLID is designed to perform collision detection in the animation, decoupling it from visualization. It keeps an internal data structure, accepting models which can be defined in the VRML97 standard[19].

2.4.4 Path Planning

The path planning problem has received considerable attention in the past. The book of laValle [91] provides an up-to-date overview on the topic. In this thesis, roadmaps are used for navigation. Roadmaps present an efficient basis for path planning, as the planning space is presented in a graph, reducing the planning problem to a path finding method. The main path finding methods are summarized in the following:

The most efficient method for path finding is **Dijkstra's Algorithm**. Most algorithms are variations of it. It solves the path finding problem without any additional information in $O((|A|+|N|)log|N|)$ steps by using a binary heap, where $|A|$ is the number of arcs and $|N|$ is the number of nodes in a network. Dijkstra's algorithm is computationally expensive for complex graphs, as it requires searching the whole graph in order to find the solution.

The **Best-First-Search (BFS) Algorithm** works similar to the Dijkstra algorithm, however, it applies a heuristic for computing the distances to the goal. Rather than selecting the vertex closest to the starting point, it selects the vertex closest to the goal. BFS is not guaranteed to find a shortest path.

The **A* Algorithm and its extension D*** [147] are a generalization of Dijkstra's algorithm, reducing the size of the subgraph to be explored as soon as additional information providing a lower-bound on the dis-

[19] G. Boll, R. Canny, and C. Martin, VRML97: The Virtual Reality Modeling Language: http://www.vrml.org/Specifications/VRML97, 1997.

tance to the target is available. A* is the most popular choice for path finding, because it is fairly flexible and can be used in a wide range of contexts. A* is comparable to other graph-searching algorithms as it is capable of searching a huge area of the map.

Mazer et al. [109] present a path planning method called the **Ariadne's Clew Algorithm**. It is designed to find paths in high-dimensional continuous spaces and applied to robots with many DoFs in static as well as dynamic environments with moving obstacles. The algorithm comprises two sub-algorithms posed as optimization problems, called *SEARCH* (target search) and *EXPLORE* (building of accessible space), applied in an interleaved manner. It enables fast planning in high-dimensional spaces.

In dynamic environments, real-time replanning is a prerequisite for safe motion under changing conditions. A framework for replanning is presented by Brock et al. [26], applicable in high-dimensional configuration spaces. In multi-robot work cells, powerful motion planning and motion execution paradigms are necessary. Usage of elastic strips enables real-time obstacle avoidance [27] and implicit motion coordination for multiple robots in a shared work space. In this work, the motion plans are augmented with a reactive component, allowing an avoidance of obstacles moving unpredictably and unexpectedly.

2.5 Developments Extending the State of the Art

Various approaches cover subsets of the sensor-based exploration problem. In order to enable autonomous operation, design rules for a sensor system must be derived in simulations. Based on the specification, suitable mechatronic hardware, i.e. an exploration sensor, must be developed. Further, methods from computational geometry must be modified to appeal for non-ideal sensing. Lastly, a general measure for information is required to serve as a task-specific driving force for exploration processes.

To date, a comprehensive approach enabling robots with complex kinematics to perform efficient exploration and inspection methods in work cells under consideration of sensing uncertainties has not been developed. No matter the task it performs - the robotic system must always observe and include information on its environment, ideally continuously maximizing the amount of knowledge available on its surroundings.

A method which combines specific task-directed actions and a secure exploration into one harmonized process is missing yet. Further, sensors capable of performing differential tasks coinstantaneously do not exist; thus, although solitary exploration and object inspection processes are well-established, a simultaneous performance of these tasks

is not possible. Possibilities for maximizing the dependability of the applied environment models by means of dedicated data fusion have not been explored to their full extent. Lastly, multi-scale approaches to environment modeling, heeding the interdependency of scale/resolution of the acquired information and acquisition time, implementing a goal-oriented trade-off between the two for each specific process, are yet to be optimized.

This thesis offers an novel approach that emanates from the current State of the Art, and the above considerations: A multisensory system integrates versatile sensors and targeted modeling procedures, allowing complex robotic systems to efficiently perform exploration and dependable environment modeling as well as specific task execution simultaneously. The approach (method and implementation) presented in this thesis combines imprecise sensing in sensor-based exploration of a physical space (under consideration of the planning space) with general view planning in this work space. The proof of concept is given by simulations and real experiments.

3

Exploration of Robot Work Space

This chapter examines the exploration problem with the objective of increasing the knowledge required for a maximal maneuverability of the robot. In unstructured, partially unknown environments, an exploration of the movability of the robotic system is required. While the mobile robot domain usually deals with systems with trivial geometry and kinematic constraints, the $\mathcal{C}$-space of articulated robots is high-dimensional, making a complete computation often intractable. In motion planning, which is usually performed in the work space $\mathcal{C}$, the environment is assumed ideal and completely known. In sensor-based motion planning, the complete knowledge of the work space is not required, but information acquired by sensors is still assumed ideal. In order to handle uncertainties while applying these motion paths to real systems, the application of safety margins is common practice.

The sensings are performed in the physical space $\mathcal{P}$. Therefore, the interrelation between work space and physical space is described. Sensor models for planning views as well as models for measurement integration are presented. Three main update rules are described: Bayes, Belief, and Fuzzy. These approaches relax the assumption of perfect sensing, which is denoted naïve update in this section. Additionally, uncertainty information is used in the planning stage, enabling more precise view planning. In order to obtain *safe-for-motion* regions, a multi-sensor strategy is applied. Even though motion uncertainty is not considered in this thesis, the environment model allows an integration of uncertain localization, based on occupancy grid maps. The measures for guiding (or driving) the exploration based on uncertain information are presented.

Figure 3.1: The DLR humanoid robot *JUSTIN*. The robot has 43 DoF. It is used for performing two-handed manipulation, object recognition, and grasp planning. A multisensory head as described in Chap. 7 is used for environment acquisition.

3.1 Problem Statement

While mobile robots usually plan in $\mathcal{C}$-spaces of dimension 3, i.e. 2-D location x and 1-D orientation θ, articulated robots typically have $\mathcal{C}$-space dimensions ≥ 6, depending on the number of DoFs of the robot. In case of humanoid robots, this may easily exceed the above numbers. The DLR two-arm robot JUSTIN [121], shown in Fig. 3.1, has a total of 43 DoF, resulting in a $\mathcal{C}$-space dimensionality of 43.

3.1.1 Exploration Task

In sensor-based motion planning, *safe-for-motion* and *interesting-for-planning* areas must be carefully distinguished. The most naïve approach to exploration is random walks, i.e. sampling unbiased views and executing scans, which might appeal to mobile robots due to their simplicity. For kinematically more complex robots, the random walk does not produce satisfying results; e.g. although the method may find a solution, the time needed for success is unacceptable.

In the sensor-based case, it is important to point out that, even in static environments, the environment map constantly changes in relation to the robot. Measuring changes the environment; literally, the known environment incrementally enlarges as more information becomes available.

Most exploration techniques employ a greedy algorithm, i.e. they aim to find the best possible outcome in the next step. This frontier-based approach is well-suited for application in the exploration problem. The goal is to gain information, but as soon as this information is acquired and processed, the robot encounters a completely new state of envi-

ronment, therefore new actions become necessary. The view planning is highly reactive in the sense of changing plans when new information becomes available. The view plan is optimal up to the first step, multi-step planning is nearly impossible. Still, especially when using sensors with small FoVs, multiple best views can be fused into an optimal scanning trajectory.

In general, the following goals for exploration can be assigned:

- $\mathcal{C}$-space exploration, i.e. in terms of entropy $H(\mathcal{C}) \rightarrow 0 \Rightarrow \mathcal{C}_{unk} \rightarrow 0$;
- $\mathcal{P}$-space exploration, i.e. $\mathcal{P}_{unk} \rightarrow 0$;
- goal exploration, i.e. $H(Q = q_{goal}) = 0$;
- object exploration, i.e. inspection, object recognition;

The goal exploration problem is a special case of the $\mathcal{C}$-space exploration. It applies a probabilistic measure for multi-step planning: The goal cannot be explored in one step, therefore the entire exploration is directed towards the goal. The complete $\mathcal{C}$-space exploration method merely plans the next step, acquires a sensing, and computes the NBV in the changed environment.

In order to gain knowledge of the environment, a task-specific view planning is required. The task is to change unknown regions into known ones. The choice of the NBV methods consequently depends on the exploration of the environment model and the following properties:

- definition of *safe-for-motion* areas;
- generative or erosive approach to sensing, i.e. surface vs. voxel-based methods.

In this thesis, voxel-based models are used, as they simplify the view planning process for work and physical space exploration. In the following, relevant notations are introduced. In order to provide a *safe-for-motion* area, a probabilistic environment model is applied, requiring the definition of suitable sensor models.

3.1.2 Notations

In the following, scalar variables are printed in regular font. Random variables (r.v.) are denoted with capital letters Z, whereas their realizations are written in lower case z. Matrices are bold faced capital letters $\mathbf{M}$ and vectors are denoted as lower case letters in bold $\mathbf{v}$. In each configuration $\mathbf{q} \in \mathcal{C}$, the robot occupies a volume $A(\mathbf{q}) \subset \mathcal{P}$. The sensor FoV in configuration $\mathbf{q}$ is denoted $\mathcal{V}(\mathbf{q})$, the subscripts unk (unknown), $free$ (free), and occ (occupied) define the states of the sets.

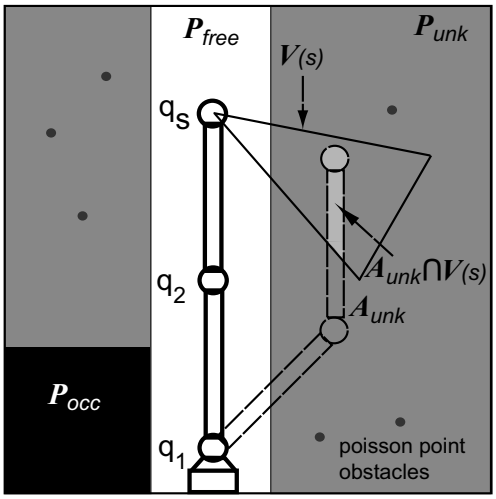


Figure 3.2: Notations for the exploration problem. The variable $\mathcal{A}$ denotes the volume of the robot, $\mathcal{P}$ is the physical space, the subscripts denote the states unk = unknown, $free$, and occ = occupied.

The sensings s are planned in order to find the optimal measurement at s_{max}, i.e. the NBV with respect to the optimization criterion. The sensing s is described by a coordinate frame attached to the sensor's origin, it determines the sensor's pose $\mathbf{x}_s$ in $\mathcal{P}$-space. The terms $\mathcal{V}(\mathbf{q})$ describes a sensing in $\mathcal{C}$-space coordinates, $\mathcal{V}(s)$ denotes a sensing in $\mathcal{P}$-space coordinates. Both of them specify the same sensor pose if the configuration $\mathbf{q}$ considers all DoFs of robot and sensor, i.e. $\mathbf{q} = [\mathbf{q}_{robot}, \mathbf{q}_{sensor}]$, which is generally assumed in this thesis. While the mapping of $\mathcal{V}(\mathbf{q})$ and $\mathcal{V}(s)$ is one-to-one for non-redundant robots, merely the forward procedure $\mathcal{V}(\mathbf{q}) \rightarrow \mathcal{V}(s)$ is unique, while the inverse procedure is ambiguous for redundant robots.

The intersection $\mathcal{A}_{unk}(\mathbf{q}) \cap \mathcal{V}_{unk}(s)$ is called the $\mathcal{C}$-zone of a configuration $\mathbf{q}$ and is denoted $\chi_{unk}(s) \in \mathcal{C}$. A beam sensor is denoted 1-D sensor (a realization might be a laser pointer), 2-D sensors acquire a single line, whereas 2.5-D sensors such as stereo sensors measure a depth field in 2-D. The known part of the sets are denoted $known$, i.e. the known part of the physical space $\mathcal{P}_{known} = \mathcal{P}_{free} + \mathcal{P}_{occ}$.

3.2 Work Space Exploration Algorithms

The $\mathcal{C}$-space entropy, introduced in [192] as a measure of ignorance of $\mathcal{C}$-space, mathematically assumes the $\mathcal{C}$-space to be a collection of n random variables, $Q_j, j = 1, .., n$ representing the status of each discretized or randomly sampled robot configuration $\mathbf{q}_j$ as being free ($Q_j = 0$) or in collision ($Q_j = 1$). An introduction to $\mathcal{C}$-space entropy and MER criterion is given in Sec. 2.1.7. This method assumes a stochastic world model, the Poisson Point Process [149, 74], which is described

in the following section. Then, different exploration methods are presented. Concluding this section, the different planning sensor models are introduced.

3.2.1 Stochastic Model of the Physical Space

The homogeneous Poisson Point Process is characterized by randomly, independently distributed points in space [149]. For motion planning, these points are considered obstacles in $\mathcal{P}$. The density of the obstacle distribution is denoted by λ. Given λ, the probability of an unknown region $\mathcal{B} \subset {\mathcal{P}_{unk}}^i$ being free of obstacles is denoted as $p(\mathcal{B}_{unk}|{\mathcal{P}^i}_{known})$ and called void probability. The void probability is equal to the unconditional probability $p(\mathcal{B}_{unk})$ of $\mathcal{B}_{unk}$ being free, which is given by:

$$p(\mathcal{B}_{unk}) = e^{-\lambda|\mathcal{B}_{unk}|} \tag{3.1}$$

This leads to the void probability of an unknown configuration $\mathbf{q}$:

$$p(\mathbf{q}|{\mathcal{P}^i}_{known}) = \begin{cases} e^{-\lambda|{\mathcal{A}^i}_{unk}(\mathbf{q})|} & \mathbf{q} \in {\mathcal{C}^i}_{unk} \\ 0 & \mathbf{q} \in {\mathcal{C}^i}_{occ} \\ 1 & \mathbf{q} \in {\mathcal{C}^i}_{free} \end{cases} \tag{3.2}$$

In this work a set of $n \in \mathbb{N}$ configurations $\mathbf{q}_1, ..., \mathbf{q}_n$ is considered a representation of $\mathcal{C}$. Due to the high complexity of the $\mathcal{C}$-space, a full computation is not possible. Hence, the set can be obtained by discretization or sampling of $\mathcal{C}$. In this thesis, the generation of possible view configurations is obtained through sampling a discretized $\mathcal{C}$-space.

3.2.2 View Planning Methods

In this section, the different planning methods published in [86, 180] are briefly presented. All of them comprise geometrical and probabilistic components, though assuming ideal sensing. The methods and their optimization goal as well as a short description of the implications are listed below:

- MER: Maximal expected Entropy Reduction criterion

$$s_{max} = \underset{s}{argmax}\ \widetilde{ER}(s) = \underset{s}{argmax} \sum_{\mathbf{q} \in \chi_{unk}(s)} er_{\mathbf{q}}(s) \tag{3.3}$$

  This method maximally reduces the $\mathcal{C}$-space entropy, induced from the Poisson Point Process [192].

- MCZV: Maximal C-Zone Volume criterion [180]

$$s_{max} = \underset{s}{argmax} \sum_{\mathbf{q} \in \chi_{unk}(s)} 1 \tag{3.4}$$

MCZV assumes that a configuration is partially unknown and intersects the sensor FoV. Not considering the complete unknown volume of the robot, the criterion simply sums up unknown configurations. Compared to MER, the method tends to maximize the volume of the unknown $\mathcal{C}$-zone of the physical space within each view.

- MCFV: Maximal $\mathcal{C}$-Free Volume criterion [180]

$$s_{max} = \underset{s}{argmax} \sum_{\mathbf{q} \in \chi_{unk}(s)} \delta(\mathcal{A}^i{}_{unk}(\mathbf{q}) \subseteq \mathcal{V}_{unk}(s)); \quad \delta(e) = \begin{cases} 1; \text{ if } e \text{ is true} \\ 0; \text{ else} \end{cases} \tag{3.5}$$

MCFV explores intersections of the unknown robot volume with the sensor FoV, omitting the unknown part of the robot which cannot be seen by the sensor.

- MPV: Maximal Physical Volume criterion [86]

$$s_{max} = \underset{s}{argmax} \sum_{s \rightarrow \mathbf{q} \in \mathcal{C}_{free}} \mathcal{V}_{unk}(s) \tag{3.6}$$

MPV aims at maximizing the unknown physical space volume that becomes known in each view. The corresponding $\mathcal{C}$-space exploration algorithm is MCFV. The MPV method is used for comparison reasons. This method merely reduces $\mathcal{P}_{unk}$, without considering the information gain for the $\mathcal{C}$-space [180].

By comparison of exploration results of the purely $\mathcal{P}$-space-based MPV approach with the MER criterion, Yu et al. [192] demonstrate that the information gain must also be defined in $\mathcal{C}$-space for its efficient exploration. Clearly, this gain must be mapped to $\mathcal{P}$-space for two reasons:

1. views are computed and performed in the physical space;
2. the information gain must be consistently represented for multiple tasks, i.e. exploration of goal configuration and objects.

The MER and MPV methods are implemented in the simulation environment for comparison reasons in order to validate the planning methods considering sensor noise, as developed in this thesis.

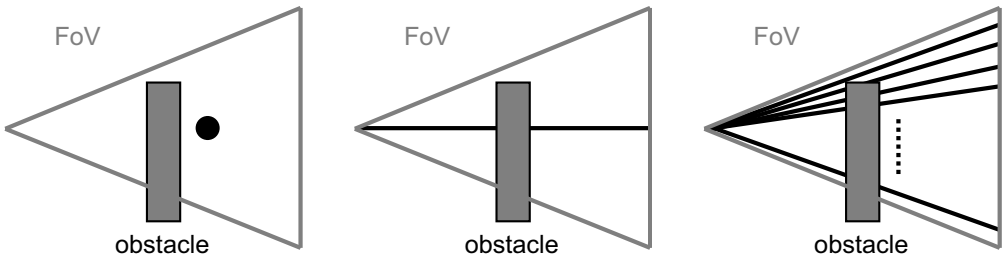
3.2.3 Geometric Sensor Models for View Planning

The methods described in the previous section require geometric sensor models for computing the set intersections, e.g. $\mathcal{A} \cap \mathcal{V}$. While the definition of the method is independent of the geometric sensor model, the resulting computation of s_{max} notably depends on the geometric sensor model in terms of computational effort. For some geometric

models, closed-form solutions exist, others require approximations. The geometric models considered in this thesis are denoted point, beam, and generalized. They are depicted in Fig. 3.3. These models are used for planning the views, they must be differentiated from the measurement model (cf. Sec. 2.1.2), which is applied beam-based.

The Point(P) geometric model examines a single point per view in the planning stage, ignoring occlusion. It corresponds to a scan-based measurement model (cf. Sec. 2.1.2). The Beam(B) model considers one ray cast per view in the planning; the Generalized(G) model approximates the sensor field of view by n beams, depending on the sensor or planning resolution. While the latter is the most realistic planning model, the computation is far more complex than for the other two.

Both Beam and Generalized sensor models account for occlusion and limited visibility. A grazing angle, as proposed in [60] and [18], is not considered. The grazing angle is defined by the normal of the surface in relation to the viewing direction, considering a limited visibility of the surface at large angles.



(a) Point planning model (P)

(b) Beam planning model (B)

(c) Generalized planning model (G)

Figure 3.3: Geometric sensor models for view planning.

3.3 Sensor Models for Measurement Interpretation

In this section, sensor models for integrating sensings, i.e. performing a map update, in the physical space are introduced. These models depend on the update rule. When choosing Bayes' Update, the inverse sensor model is required. For other update rules such as Fuzzy and Belief, the forward model is required. In the following, details of the models, including parameters and their estimation, are described.

In general, beam-based models are chosen for updating as their interpretation is based on the physics of the sensor. Landmarks are neither considered nor required in exploration. Scan-based methods utilize 2-D likelihood maps, which have to be matched. This approach is well-suited for 2-D projections of the environment, as predominant

in mobile robotics. In this thesis, the approach is not limited to 2-D but extended to 3-D environments, making scan-based sensor models less applicable. Usually, map-matching or likelihood fields are used in the scan-based case. Map-matching is computationally expensive in 3-D.

3.3.1 Naïve Sensor Model

The naïve model assumes that range measurements are correct. Therefore, no interpretation of the measurements is performed. This model relates to the ideal sensing assumption commonly applied in sensor-based motion planning. In this work, the naïve model is used for comparison of the planning methods in order to emphasize the necessity of sensor interpretation.

3.3.2 Inverse Sensor Model

The inverse sensor model required for the Bayes' Update. The map of the environment, i.e the map of the physical space $\mathcal{P}$, is assumed to consist of discretized cells c_i, which are cubic. The robot's pose is denoted $\mathbf{x}$, the sensor pose $\mathbf{x}_s$, which is obtained by $\mathbf{x}_s = {}_s\mathbf{T}^r \cdot \mathbf{x}$, where $\mathbf{T}$ is the calibration of the sensor on the robot. In case of a voxel space representation of $\mathcal{P}$, each cell c_i has the same volume, in an octree representation as detailed in Chap. 5, the volume of the cell is variable but lower bounded by the resolution. In 3-D spaces, the volume of the cell increases by $2^{3 \cdot k}$, with $k \in \mathbb{N}$ being the level of the octree.

Each cell c_i represents a binary random variable, which can have the state $c_i = 1$, i.e. *occupied* and $c_i = 0$, i.e. *not occupied*. The measured value d is the realization of a random variable D. This leads to

$$p(D = d | c_i = 1, \mathbf{x}_s) \tag{3.7}$$

representing the probability density for obtaining a measurement d, given that c_i is occupied and the sensor is located at $\mathbf{x}_s$. No error in $\mathbf{x}_s$ is considered. The probability density for obtaining d at $\mathbf{x}_s$, given c_i is free, is denoted as follows:

$$p(D = d | c_i = 0, \mathbf{x}_s). \tag{3.8}$$

Only cells inside the sensor FoV are taken into account. In order to adequately consider the sensor properties, possible sources of errors are identified. In general, each sensor has a limited FoV, defined by a minimal and maximal sensing range: $d \in [d_{min}, d_{max}]$. In addition, the opening angles complete the FoV definition. In range direction of a beam, d_{c_i} denotes the distance of the cell c_i from the origin of the respective sensor. Furthermore, perfect and flawed surfaces must be dealt with. The latter are specular (e.g. polished metal, mirrors), light

adsorbing (e.g. black-colored objects), or light-transmissive (e.g. glass), where triangulation-based optical sensors produce undesired results.

Model for Occupied Cells

Perfect surfaces at c_i are prone to three sources of error:

1. impreciseness of the sensor;
2. cell incorporating a perfect surface in between the sensor origin and the current cell c_i in view direction;
3. flawed surface closer to the sensor than c_i.

The sensor inaccuracy is usually modeled as Gaussian distribution, its expectation being the real distance to the surface, while the variance is the inaccuracy of the sensor. Including the restricted FoV, this leads to:

$$p_1(D = d|c_i = 1, \mathbf{x}_s) = \eta_1 \cdot exp(-\frac{(d - d_{c_i})^2}{\sigma^2(d_{c_i})}) \cdot I_{\{d_{min}<d<d_{max}\}} \tag{3.9}$$

with the scale η_1 and distance-dependent variance $\sigma^2(d_{c_i})$. Behind perfect surfaces at d_{c_i}, seen in direction of sensing, a measurement is very unlikely. In front of perfect surfaces, there is a possibility of another perfect surface. The farther a cell is located from the sensor origin, the more likely is its occlusion by a perfect surface, which leads to the following exponential distribution:

$$p_2(D = d|c_i = 1, \mathbf{x}_s) = \eta_2\lambda_2 exp(-\lambda_2 d) \cdot I_{\{d_{min}<d<d_{c_i}\}} \ . \tag{3.10}$$

Again, η_2 is the scale. Finally, the existence of a flawed surface in front of the measured cell must be modeled. A statement on how the measurement is affected is not possible, therefore a uniform distribution in $[d_{min}, d_{max}]$ is applied for this situation[1]. The probability of the sensor encountering a flawed surface in the line of sight increases with the cell's distance from the origin of the sensor.

$$p_3(D = d|c_i = 1, \mathbf{x}_s) = \frac{\kappa(d_{c_i})}{d_{max} - d_{min}} \ . \tag{3.11}$$

In Eq. (3.11), $\kappa : [d_{min}, d_{max}] \rightarrow [0, 1]$ is monotone increasing with d_{c_i} and $\kappa(d_{max}) < 1$. The latter requirement considers that no measurement is obtained beyond d_{max} or by absorption of the laser. The overall model for a perfect surface can be defined as a mixture

$$p_p(D = d|c_i = 1, \mathbf{x}_s) = \sum_{i=1,..,3} w_i p_i, \quad \sum_{i=1,..,3} w_i = 1 \wedge w_i \in \mathbb{R}^+ \ . \tag{3.12}$$

[1] A common method for representing influences without clear assumption on the effect is the uniform distribution.

For further details on sensor models, the reader is referred to [84] and [164]. In this thesis, the models in [164] are modified in the sense of adding κ in Eq. (3.11) for the uniform distribution, which models the amount of outliers as well as the dependance on the sensing distance. This situation is not considered in [164].

Model for Free Cells

The model for free cells is developed analogously to the derivations for the occupied cells. It is assumed that no obstacle is present in free cells, therefore the probability p_1 in Eq. (3.9), modeling Gaussian noise on surfaces, must not be considered. This leads to the following equation:

$$p_p(D = d|c_i = 0, \mathbf{x}_s) = (w_2\eta_2\lambda_2 exp(-\lambda_2 d) + w_3\frac{\kappa(d_{c_i})}{d_{max} - d_{min}}) \cdot I_{\{d_{min}<d<d_{c_i}\}} \ . \tag{3.13}$$

Complete Model for Free and Occupied Cells

In order to model the sensor properties completely, the influence of flawed surfaces must be taken into account. The derivations above consider perfect surfaces, whereas flawed surfaces cannot be modeled by Gaussian noise. Flawed surfaces may be specular or absorbing, therefore they are modeled by a weighted summation of exponential and uniform distributions. This leads to

$$p_f(D = d|c_i = 1, \mathbf{x}_s) = (w_2\eta_2\lambda_2 exp(-\lambda_2 d) + w_3\frac{\kappa(d_{c_i})}{d_{max} - d_{min}}) \cdot I_{\{d_{min}<d<d_{d_i}\}} \ . \tag{3.14}$$

in analogy to the derivations above, omitting the Gaussian distribution.

In order to combine flawed and perfect surfaces into one model (cf. Eq. (3.14) and Eq. (3.12)), a mixed distribution regarding the respective influence of the surface characteristics is reasonable. In the following, this mixed distribution for perfect and flawed surfaces is derived. First, the model for $c_i = 1$ is stated as

$$\begin{aligned} p_m(D = d|c_i = 1, \mathbf{x}_s) = &(1 - P(S))p_p(D = d|c_i = 1, \mathbf{x}_s) + P(S)\cdot \\ &p_f(D = d|c_i = 1, \mathbf{x}_s) \end{aligned} \tag{3.15}$$

For free cells, the probability

$$p_m(Z = z|c_i = 0, \mathbf{x}_s) = p_p(Z = z|c_i = 0, \mathbf{x}_s) \tag{3.16}$$

is computed in analogy. In the above equations, $P(S)$ denotes the probability for a flawed surface. A priori, $P(S) \equiv 0$ for each cell.

The estimation of $P(S)$, the probability for a flawed (specular) surface, is not trivial. In [84], $P(S)$ is evaluated by a heuristic; the computation of $P(S)$ is derived in Sec. 3.4.3.

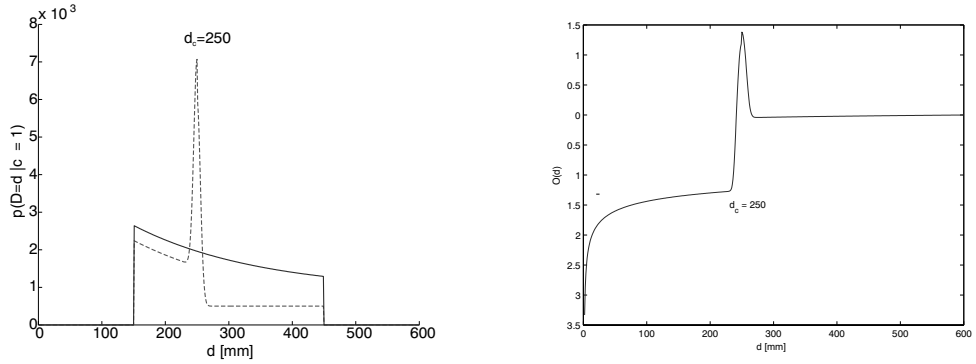


Figure 3.4: Complete models. *Left:* Model for occupied cell (dotted red) at distance $d = 250mm$ and model for empty cell/flawed surface (blue). *Right:* Typical plot of the log-odds.

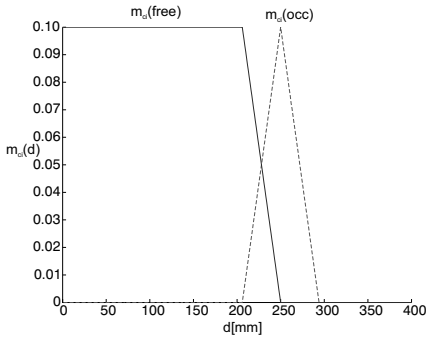


Figure 3.5: Induced belief in occupancy (dotted red) based on a measurement ($d = 250mm$) and free space of a cell for the parameters $\nu = 0.1$ and $\zeta = 44$. The free space and occupancy hypothesis overlap, therefore contradiction is induced in the update method.

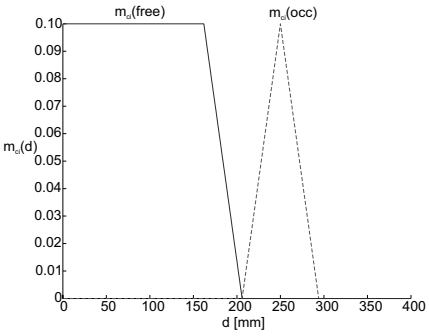


Figure 3.6: Induced belief in occupancy (dotted red) and free space (blue) of a cell based on a measurement ($d = 250mm$) for the parameters $\nu = 0.1$ and $\zeta = 44$. In this case, the free space and occupancy belief do not overlap, reducing contradiction in the update.

3.3.3 Forward Sensor Model

The forward sensor model used in Belief and Fuzzy Update of the map is described in this section. For both update types, the sensor model is identical. Given a measurement d and d_{c_i}, the distance of a cell c_i from the sensor (only cells intersecting the sensor ray are considered), the belief (or membership) in occupancy $m_{c_i}(occ)$ and vacancy $m_{c_i}(free)$ is derived. In the the following equations, ν denotes the maximal induced belief (membership), ζ the sensor-dependent width of a region around d.

$$m_{c_i}(free) = \begin{cases} \nu, & d_{c_i} \leq d - \zeta; \\ (d - d_{c_i})\frac{\nu}{\zeta}, & d - \zeta \leq d_{c_i} \leq d; \\ 0, & \text{else.} \end{cases} \tag{3.17}$$

$$m_{c_i}(occ) = \begin{cases} (d_{c_i} - d - \zeta)\frac{\nu}{\zeta}, & d - \zeta \leq d_{c_i} \leq d; \\ (d + \zeta - d_{c_i})\frac{\nu}{\zeta}, & d \leq d_{c_i} \leq d + \zeta; \\ 0, & \text{else.} \end{cases} \tag{3.18}$$

The corresponding plots for Eq. (3.17) and Eq. (3.18) are shown in Fig. 3.5. The belief (membership) $m_{c_i}(free)$ and $m_{c_i}(occ)$ overlap in the sensor model and are both updated for $d - \zeta < d_{c_i} < d$. This leads to a high contradiction in Fuzzy and in Belief Update. Therefore, another model (depicted in Fig. 3.6) is given:

$$m_{c_i}(free) = \begin{cases} \nu, & d_{c_i} \leq d - 2 \cdot \zeta; \\ (d - d_{c_i})\frac{\nu}{\zeta}, & d - 2 \cdot \zeta \leq d_{c_i} \leq d - \zeta; \\ 0, & \text{else.} \end{cases} \tag{3.19}$$

$$m_{c_i}(occ) = \begin{cases} (d_{c_i} - d - \delta)\frac{\nu}{\zeta}, & d - \zeta \leq d_{c_i} \leq d; \\ (d + \zeta - d_{c_i})\frac{\nu}{\zeta}, & d \leq d_{c_i} \leq d + \zeta; \\ 0, & \text{else.} \end{cases} \tag{3.20}$$

The reduction of overlap merely changes the definition of $m_{c_i}(free)$ as presented in Eq. (3.19). The formulation of Eq. (3.20) remains identical to Eq. (3.18).

3.4 Physical Space Update based on Sensor Models

There are multiple ways of handling sensor uncertainties in robotics. Most approaches, e.g. Elfes [47] and Konolige [84], apply Bayes' Update to obtain maps of a robotic environment. Others use Belief approaches (e.g. Pagac and Durrant-Whyte [123], as well as [46, 167, 193]). Fuzzy Update is less common, Gambino et al. [56] apply it for a stereo sensor. An overview of the approaches is published in [46].

3.4.1 General Remarks

Bayes' Rule is profoundly based on the probability theory, its use is least controversial in robotics, while the use of Fuzzy Theory is heavily discussed in the community. Fuzzy's main advantage is the free definition of membership functions and combination rules, which allows a good adaption on the problem. This advantage is also the major criticism of the approach, as it requires more knowledge and consideration while defining membership functions and combination rules (t-Conorms and t-Norms). This makes the approach appear to be 'trial-and-error'-based. The largest advantage of Fuzzy is that sensings do not need to be independent when updating grid-based maps, as required in Bayes' and Belief Theory. In robotics, the requirement of independent measurements is often neglected, actually violating the theory.

In order to implement a common approach for the exploration and inspection task, the entropy calculation for $\mathcal{C}$-space exploration needs to

be adapted to the sensor noise. By considering sensor noise, a statistical measure for the physical space is gained, the Poisson Point Model might be replenished. The use of the sensor model relates more to the physics of the environment than the Poisson Point Model. However, it causes notably higher computational cost. The closer an object is to the sensor, the smaller the variance in the detection of an obstacle becomes. No information about objects outside the FoV of the sensor can be obtained. It must be taken into account that the detection of an obstacle reduces the entropy massively. Knowledge of a configuration being in collision is more informative than knowing that $\mathcal{A}(\mathbf{q}) \cap \mathcal{V}(s)$ is free. The latter needs knowledge of $\mathcal{A}_{unk}(\mathbf{q})$ to decide about $\mathbf{q}$ being collision-free.

The approach of map-based modeling of sensor uncertainty, limited to grid-based maps and not considering feature-based approaches, is derived following the occupancy map theory, as published by Elfes [47]. In general, occupancy grids can be interpreted stochastically as Markov Random Fields (MRF) of order 0; the independent cell states can be estimated as independent random variables. A computationally more expensive approach of calculating higher order MRFs leads to a frequency approach (Boolean Model [149]). Besides the normalization and consistency error [47], another drawback of the occupancy grid approach is the assumption of statistical independence of the cells. However, as already discussed, if all joint probabilities were to be computed, the computational effort would be $n!$ if the 2-D world model had n-cells, making the computation intractable.

Therefore, the update of the map is performed with Log-Odds [84] (cf. Sec. A.2). In the following, the update process is described for an occupancy grid implemented as voxel space. The extension to octree representation is described in Chap. 6. The occupancy grid is defined as a 3-D map in physical space, i.e. cells are assumed to be axis-aligned allowing to neglect their orientation. The grid's resolution, i.e. the cell size, is denoted by res. The variable res depends on the sensor used and the accuracy applied for motion planning. The following nomenclature is used when calculating the probability of a cell being free (=0) or occupied (=1):

$$\begin{aligned} c_i(x,y,z) &\equiv [c_i(x,y,z) = 1] \\ \overline{c}_i(x,y,z) &\equiv [c_i(x,y,z) = 0] \ . \end{aligned} \tag{3.21}$$

For Bayes' Update, the inverse sensor model based on the physical properties derived in Sec. 3.3.2 is required. Literally, the sensor model needs to be applied to all world configurations. To simplify the approach, the variable d is expressed by the cell values for $c_i(x)$ (or $c_i(\mathbf{x})$ in the multi-dimensional case). For 1-D, this means $d = x$. The term $p(d)$ contains the information on the affected cells inside the sensor FoV $\mathcal{V}(s)$ for each scan s. This procedure must be applied to all cells in the grid in order to obtain the resulting occupancy map, making

the general application of Bayes' Rule intractable. Therefore the assumptions described in Sec. 3.4.2 are made to obtain an incrementally suitable update method.

3.4.2 Definition of Measurements

The following derivations assume a one-dimensional sensing beam. The sensor is located at $\mathbf{x}_s = [\mathbf{x}_{loc}; \mathbf{x}_{dir}]$ with $\mathbf{x}_{loc}$ being the location and $\mathbf{x}_{dir}$ being the orientation, i.e sensing direction, of the beam. The distance in which a measurement is acquired is denoted d. The precision of the sensor is used to model the variance required for the inverse model as well as the dimensions of the region around d in the forward model.

3.4.3 Bayes' Update

For Bayes' Update, the inverse model derived in Sec. 3.3.2 is used. In this section, special attention is given to the computation of $P(S)$, i.e. the probability for a flawed surface, as introduced in [84].

In general, the Odd is defined by

$$O(A|B) = \frac{P(A|B)}{P(\bar{A}|B)} \tag{3.22}$$

and the Likelihood Quotient

$$LQ = \frac{P(A|B)}{P(A|\bar{B})} \ . \tag{3.23}$$

The logarithm of the LQ is denoted Likelihood Ratio (LR). As for Bayes' Update the following formulation

$$L(D_j = d_j|c_i = 1)) = \frac{p(D_j = d_j|c_i)}{p(D_j = d_j|\bar{c}_i)} \tag{3.24}$$

is used, Likelihood Quotients instead of Odds are computed. However, the terms Odd and Likelihood Quotient are used synonymic in this thesis.

All Log-Odds for occupied cells, L_{occ}, are computed using Eq. (3.12), Eq. (3.13), and the definition of the Odd as follows:

$$L(D_j = d_j|c_i = 1, \mathbf{x}_j) := \frac{p(D_j = D_j|c_i = 1, \mathbf{x}_j)}{p(D_j = D_j|c_i = 0, \mathbf{x}_j)} \ . \tag{3.25}$$

The Log-Odd, i.e. $l = \log L$ is computed by logarithmizing the Odd. If the sum of all Log-Odds is larger than a threshold[2] th, $P(S) = 1$. If

[2]For the selection for th, [84] propose a value between 2 and 3. In this thesis, $th = 3$ is used unless otherwise stated. The larger th, the more slowly a cell is considered specular, smaller values of th result in an earlier assumption of specularity.

the sum is smaller than 0, $P(0) = 0$. In between these values, a linear interpolation is performed:

$$P(S) = \begin{cases} 0 & \sum l_{occ} < 0 \\ \frac{\sum l_{occ}}{th} & 0 \leq \sum l_{occ} \leq th \\ 1 & \sum l_{occ} > th \end{cases} \tag{3.26}$$

The weighting with $P(S)$ and $1 - P(S)$ allows a computation of l_{free} as described in Eq. (3.15), Eq. (3.16), and Eq. (3.25), and its addition to the earlier calculated l_{occ}. This derivation requires a new computation of L_{free} whenever a new measurement in the cell is performed.

Both L_{free} and L_{occ} are computed and multiplied using Eq. (3.12) and Eq. (3.13), $P(S)$ is estimated as derived above. Then

$$l_{full} = log(L_{occ}) + log(L_{free}(1 - P(S)) + P(S)) \tag{3.27}$$

is used as an approximation, allowing an incremental update.

The individual estimation of $P(S)$ for each cell is the major improvement described in [84] to detect measurements on specular surfaces by using conflicting information. A global computation of $P(S)$ would not be an improvement, as all cells would be treated identically. While Konolige [84] uses an ultrasonic sensor with notable on specular surfaces, caused by measured distances exceeding the true value (too many cells are assumed to be free beyond the true obstacle), the impact of specularity on a laser-scanner is different. In the presence of specular surfaces, laser sensors usually measure values that are too low, a fact which originally conflicts with the assumption that specular cells are preferable obstacles. With regard to *safe-for-motion* areas, this approach is nevertheless justifiable, as it is a conservative approach ensuring safe motion in the presence of specular surfaces.

3.4.4 Belief Update

A measurement d induces the existence of a surface and the lack of occupied cells in between this surface and the sensor origin in measurement direction. This means that the measurement only induces belief in occupancy in vicinity to d. A belief in non-occupancy is solely induced on cells closer to the sensor.

The update of m is exemplarily described by $m(\{occ\})$[3]. Given a measurement d and d_{c_i}, the distance of cell c_i from the sensor (only cells hit by the beam are considered), the belief in occupancy of a cell c_i, $m_{c_i}(occ)$, and the belief in non-occupancy, $m_{c_i}(free)$ are expressed in Eq. (3.17) and Eq. (3.18) and depicted in Fig. 3.5, with overlap between $m_{c_i}(occ)$ and $m_{c_i}(free)$. The model definition without overlap is formulated in Eq. (3.19) and Eq. (3.20) and depicted in Fig. 3.6.

[3]For better readability, set brackets are omitted for sets, i.e $m(occ) = m(\{occ\})$.

Separate modeling of free and occupied cells is performed, because $m_{c_i}(occ) + m_{c_i}(free) \neq 1$ is true in general. This leads to non-complementary results for *cell occupied* and *cell free*. Therefore, $m_{c_i}(occ)$ is the belief in complete occupancy of c_i and $m_{c_i}(free)$ denotes the complete emptiness of c_i, which is not possible in Bayes' Update.

For each cell, three possible states exist: *completely occupied* (occ), *empty* (free), or *neither occupied nor free* (nof). The frame of discernment, i.e. the set of all opposing hypotheses (cf. Eq. (A.19), page 190), for each cell is denoted as:

$$\Omega = \{occ, free, nof\} \ .$$

A priori, no knowledge of a cell's state is available. For $\Lambda = 2^\Omega$ this is described by

$$\forall c_i \in \mathcal{P} : \forall A \in \Lambda : m_{c_i}(A) = \begin{cases} 1 & \text{if} \quad A = \{occ, free, nof\} \\ 0 & \text{else} \end{cases} \ .$$

If a basic belief mass $m^{curr}_{c_i}(occ)$ for a cell c_i exists for occupancy and a measurement induces a base measure $m^d_{c_i}(occ)$ for the respective cell, then these two base measures are fused using Dempster's Rule of Combination. It follows from

$$m^{curr}_{c_i}(nof) = m^d_{c_i}(nof) = 0$$

and

$$m^d(occ, free) = m^{curr}(occ, free) = m^d(occ, nof) = m^{curr}(occ, nof) = 0$$

that

$$\begin{aligned} m^{new}_{c_i}(occ) :=& m^{curr}_{c_i} \oplus m^d_{c_i}(occ) = \frac{\sum\limits_{\substack{A,B\in\Lambda \\ A\cap B=occ}} m_1(A)m_2(B)}{1 - \sum\limits_{\substack{A,B\in\Lambda \\ A\cap B=\emptyset}} m_1(B)m_2(A)} \\ =& \frac{m^{curr}_{c_i}(occ) + m^d_{c_i}(occ) + m^{curr}_{c_i}(occ)m^d_{c_i}(occ)}{1 - m^{curr}_{c_i}(occ)m^d_{c_i}(free) + m^{curr}_{c_i}(free)m^d_{c_i}(occ)} \\ &+ \frac{m^{curr}_{c_i}(occ) \cdot m^d_{c_i}(occ, free) + m^{curr}_{c_i}(occ, free) \cdot m^d_{c_i}(occ)}{1 - m^{curr}_{c_i}(occ)m^d_{c_i}(free) + m^{curr}_{c_i}(free)m^d_{c_i}(occ)} \ , \end{aligned} \tag{3.28}$$

and

$$m^{new}_{c_i}(nof) = 0 \ .$$

The base measure m^{curr} is updated with the new base measure:

$$m^{curr} := m^{new}.$$

As the models do not induce a positive $m^d(nof), m^d(\{occ, nof\})$ or $m^d(\{occ, free\})$, it is true that

$$m^{curr}(nof) = m^{curr}(\{nof, free\}) = m^{curr}(\{occ, nof\}) = 0 \ ,$$

thus the update is performed based on Eq. (3.28). In analogy,

$$m_{c_i}^{new}(free) := \frac{m_{c_i}^{curr}(occ) + m_{c_i}^{d}(occ) + m_{c_i}^{curr}(occ)m_{c_i}^{d}(occ)}{1 - m_{c_i}^{curr}(occ)m_{c_i}^{d}(free) + m_{c_i}^{curr}(free)m_{c_i}^{d}(occ)} + \frac{m_{c_i}^{curr}(occ) \cdot m_{c_i}^{d}(occ, free) + m_{c_i}^{curr}(occ, free) \cdot m_{c_i}^{d}(occ)}{1 - m_{c_i}^{curr}(occ)m_{c_i}^{d}(free) + m_{c_i}^{curr}(free)m_{c_i}^{d}(occ)} \ . \tag{3.29}$$

is true.

The computation follows the reasoning in [123]. In this article, the states *occupied* and *non-occupied* are assumed to be complementary. In this thesis, laser sensors and coarse resolution compared to the sensor preciseness are used, therefore an additional state nof is introduced (*neither occupied nor free*). The computation remains identical due to the selection of the models, while it would change in case measurements induced positive belief in the sets $\{occ, nof\}, \{free, nof\}$ or $\{occ, free\}$.

3.4.5 Fuzzy Update

In this section, the procedure to update environments using fuzzy sets is described, considering two sets: the set of free cells F and the set of occupied cells O.

The update is performed incrementally. Initially, nothing is known on the set of free cells, i.e. the degree of membership for the set of free cells is zero for all cells:

$$\forall c_i \in \mathcal{P} : \mu_{free}(c_i) = 0 \ .$$

A measurement $d_j := (d_j, \mathbf{x}_j)$ induces a (positive) degree of membership for the set of free cells in the FoV , therefore a measurement-related set $F(d_j)$ is formed. The current (global) set of free cells F is updated by union with the set $F(d_j)$:

$$F := F \cup F(d_j) \ .$$

This process is performed with every new measurement and the set of free cells is induced respectively. After n measurements, F is computed by:

$$F := \bigcup_{j=1}^{n} F(d_j).$$

In analogy, the set of occupied cells O is calculated.

The membership functions are computed based on the forward sensor models derived in Sec. 3.3.3

$$\mu_{free}(x) = \begin{cases} \nu & \text{for} \quad x < d - \zeta \\ (d - x)\frac{\nu}{\zeta} & \text{for} \quad d - \zeta \leq x < d \\ 0 & \text{else} \end{cases}$$

$$\mu_{occ}(x) = \begin{cases} (x-d+\zeta)\frac{\nu}{\zeta} & \text{for} \quad d-\zeta \leq \quad x < d \\ (d+\zeta-x)\frac{\nu}{\zeta} & \text{for} \qquad d \leq \quad x < d+\zeta \\ 0 & \text{else} \end{cases} .$$

The physical space update is performed: As no knowledge of occupancy or non-occupancy is available initially,

$$\forall c_i \in \mathcal{P} : \mu_{free}(c_i) = \mu_{occ}(c_i) = 0$$

is valid.

If the membership function of the current set of occupied cells is given by m_b^{curr} and the set of occupied cells m_{occ}^d, induced by the measurement d, then the union set is computed using a t-Conorm s:

$$\forall c_i \ \in \mathcal{P} : \mu_{occ}^{new}(c_i) := s(\mu_{occ}^{curr}(c_i), \mu_{occ}^{d}(c_i))$$

and updated respectively:

$$\mu_{occ}^{curr} := \mu_{occ}^{new} .$$

The set of free cell is updated analogously.

Similarly to the Belief Approach, free and occupied cells are modeled independently. Additionally, the update is performed independently in Fuzzy. Using the Dempster-Shafer method, the update is performed via the combination rule, i.e. c_i: $m_{c_i}(occ) + m_{c_i}(free) \leq 1$ is true for all cells c_i. While modeling with fuzzy sets, a complete independence of the sets is given.

3.5 C-space Entropy Considering Sensor Noise

In this section, the ideal beam sensor model derived by Yu et al. [188] is extended for considering noise in the planning stage. The outcome is a view planning algorithm based on imprecise sensing. In sensor-based motion planning, assumptions about the environment must be made in order to enable view planning, i.e. the Poisson Point Model is applied.

An ideal sensor (not subject to noise) can be modeled as follows: The sensor measures the first obstacle at d_{obs} perfectly. Therefore, the probability for a measurement d given d_{obs} is denoted as:

$$p(d|d_{obs})_\delta = \delta(d - d_{obs}) \tag{3.30}$$

The Dirac Impulse δ models this event; exclusively at d_{obs}, the probability for an obstacle is $p(d|d_{obs}) \neq 0$.

An ideal sensor has a limited vision, a maximal sensing distance d_{max} is the upper bound for the measurement range. The minimum sensing range is denoted d_{min}. An ideal sensor senses a region free in front of

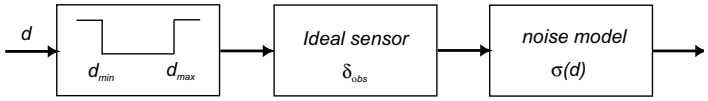


Figure 3.7: Noisy planning model based on the ideal sensor

an obstacle in sensing direction. It cannot miss the first obstacle, thus the exponential distribution used for the inverse model in Eq. (3.10) is not considered. The sensor measures the true location of the obstacle at $d_{obs} \in [d_{min}, d_{max}]$. No information on the state of the area behind an obstacle can be acquired. These considerations lead to the following model for an ideal sensor. The minimal sensing distance $d_{min} = 0$ is assumed in the following:

$$p_\delta(c_i = occ|d_{obs}, d_{c_i}) = \begin{cases} 0 & ; \ d_{c_i} < d_{obs} \\ 1 & ; \ d_{c_i} = d_{obs} \\ 0.5 & ; \ d_{c_i} > d_{obs} \end{cases} \tag{3.31}$$

where d_{obs} denotes the obstacle, also denoted *hit-point* or $hitpt$.

If there is no such obstacle (no *hit-point*) in the sensor range, the result of the ideal sensor is

$$p_\delta(c_i = occ|d_{nobs}, d_{c_i} = \begin{cases} 0 & ; \ d_{c_i} < d_{max} \\ 0.5 & ; \ d_{c_i} \geq d_{max} \end{cases} \tag{3.32}$$

If the sensor model in Eq. (7.3) is used, the calculation can be described as a convolution of the ideal sensor result with a Gaussian. This leads to the following result if the sensor is subject to noise:

$$p_G(c_i = occ|d_{c_i}) = \begin{cases} p_\delta(c_i|d_{obs}, d_{c_i}) * N(0, \sigma_d(d_{obs})) & ; \text{ hit-point} \\ p_\delta(c_i|d_{nobs}, d_{c_i}) * N(0, \sigma_d(d_{max})) & ; \text{ no hit-point} \end{cases} \tag{3.33}$$

The ideal sensor is blurred by the Gaussian. The variance $\sigma(d_{obs})$ or $\sigma(d_{max})$ depends on the true sensing distance d_{obs}, or the true maximal sensing distance d_{max} respectively. The convolution provides the solution for the noisy sensor, based on the ideal one. The process is depicted in Fig. 3.7.

If $u(t)$ denotes the Heaviside-function, the convolution is performed for

$$p(c_i = occ|d_{obs}, d_{c_i}) = \int_0^\infty \frac{1}{2}(\delta(t - d_{obs}) + u(t - d_{obs})) \cdot \frac{1}{\sqrt{2\pi\sigma(d_{obs})}} e^{-\frac{1}{2}\frac{(d_{c_i}-t)^2}{\sigma^2(d_{obs})}} dt \tag{3.34}$$

and

$$p(c_i = occ|d_{nobs}, d_{c_i}) = \int_0^\infty \frac{1}{2} u(t - d_{max}) \cdot \frac{1}{\sqrt{2\pi\sigma(d_{max})}} e^{-\frac{1}{2}\frac{(d_{c_i}-t)^2}{\sigma^2(d_{max})}} dt \ . \tag{3.35}$$

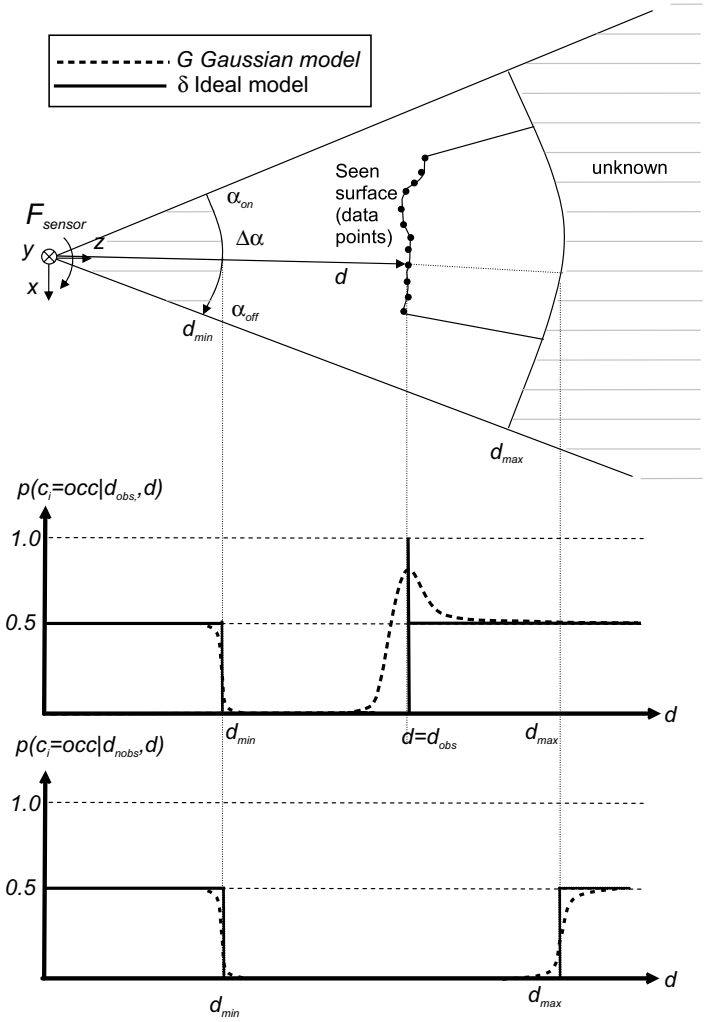


Figure 3.8: Probability distribution assuming uncertain sensing. The upper plot shows a setting with an obstacle inside the FoV, the lower plot shows the sensor model with no obstacle in the FoV. The derivations in this section assume $d_{min} = 0$ in order to simplify the calculations.

The preceding computations merely regard the probability of the existence of an obstacle at d_{obs}. The following substitutions are introduced: The probability for a cell c_i being occupied is denoted $p(c_i) = p(c_i = occ)$, the probability for a cell not being occupied is $p(\overline{c}_i) = p(c_i = \neg occ)$. The inverse likelihood of a cell being free can be derived from

$$p(\overline{c}_i|d) + p(c_i|d) = 1 \ . \tag{3.36}$$

In calculating the $\mathcal{C}$-space entropy, the probability of a cell being free is described by:

$$p(\overline{c}_i|d) = 1 - p(c_i|d) \ . \tag{3.37}$$

The environment model lacks information for virtual sensings. Therefore, assumptions on the obstacle distribution are made. For the obstacle distribution, the Poisson Point Model is consulted and combined

with results from the environment model, i.e. an occupancy grid, derived in the previous sections. A probability density function (PDF) is defined, which either describes the existence of an obstacle within the FoV or the situation that the entire FoV is free of obstacles. It is not necessary to compute the entire occupancy map, but to consider merely the results of the occupancy distribution.

Two different events for the $\mathcal{C}$-space entropy computation must be distinguished:

1. there is no hit-point in the $\mathcal{V}(s)$;
2. there is a hit-point within the FoV, at distance d_{obs} from the sensor origin.

The first situation can be derived from the occupancy grid. There is no hit-point in $\mathcal{V}(s)$, i.e. the region inside the FoV is known to be free, areas outside d_{max} are unknown. The second event, i.e. a hit-point in the sensor range, can be described analogously. The preceding computations only consider the probability of the existence of an obstacle at d_{obs}. The entropy for a configuration $\mathbf{q}$, i.e. the realization of the random variable Q, can be computed as:

$$H(Q) = \underbrace{p(\mathbf{q})logp(\mathbf{q})}_{obstacle} + \underbrace{(1-p(\mathbf{q}))log(1-p(\mathbf{q}))}_{free} \tag{3.38}$$

$$H(\mathcal{C}) = \sum H(Q) \tag{3.39}$$

The function $p(q)$ describes the probability of a region being free based on the world model, in this case the Poisson Point Model. The PDF for a noisy sensor can be described as follows:

First it is assumed that there is no obstacle within the FoV. The PDF consists of a convolution between the ideal sensor and Gaussian:

$$p(c_i = occ|\neg d_{obs}, d_{c_i}) = N(0, \sigma(d_{max})) * a \cdot u(d_{c_i} - d_{max}) \ . \tag{3.40}$$

The variable a describes the prior for an unknown region, assuming $a = \frac{1}{2}$. If there is an obstacle within the FoV, the PDF is denoted as follows:

$$p(c_i = occ|d_{obs}, d_{c_i}) = N(0, \sigma(d_{obs})) * a(u(d_{c_i} - d_{obs}) + \delta(d_{c_i} - d_{obs})) \ . \tag{3.41}$$

Enlarged in the integral form, the equations above result into

$$p(c_i = occ|d_{obs}, d_{c_i}) = a \cdot \int (u(t - d_{obs}) + \delta(t - d_{obs})) \cdot \frac{1}{\sqrt{2\pi}\sigma(d_{obs})} e^{-\frac{1}{2}\frac{(d_{c_i}-t)^2}{\sigma^2(d_{obs})}} dt \tag{3.42}$$

if an obstacle is sensed. Similarly, the PDF is computed as

$$p(c_i = occ|\neg d_{obs}, d_{c_i}) = a \cdot \int (u(t - d_{max})) \cdot \frac{1}{\sqrt{2\pi}\sigma(d_{max})} e^{-\frac{1}{2}\frac{(d_{c_i}-t)^2}{\sigma^2(d_{max})}} dt \tag{3.43}$$

if the FoV is sensed free. The solution to the convolution can be displayed as

$$p(c_i = occ|d_{obs}, d_{c_i}) = a[\frac{1}{\sqrt{2\pi}\sigma(d_{obs})} e^{-\frac{1}{2}\frac{(d_{c_i}-d_{obs})^2}{\sigma^2(d_{obs})}} + \frac{1}{2}(1+\mathbf{erf}(\frac{\sqrt{2}\cdot(d_{c_i}-d_{obs})}{2\cdot\sigma(d_{obs})}))] \tag{3.44}$$

and

$$p(c_i = occ|\neg d_{obs}, d_{c_i}) = \frac{a}{2}(1 + \mathbf{erf}(\frac{\sqrt{2}\cdot(d_{c_i}-d_{obs})}{2\cdot\sigma(d_{obs})})) \ . \tag{3.45}$$

The function $\mathbf{erf}()$ denotes the error-function, which describes the integral of a normal distribution. It can also be defined as a MacLaurin Series:

$$\mathbf{erf}(x) = \frac{2}{\sqrt{\pi}}\sum_{n=0}^{\infty}\frac{(-1)^n z^{2n+1}}{n!(2n+1)} = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt \ . \tag{3.46}$$

The information gain density (IGD) is applied whenever a geometric sensor with zero volume FoV is considered:

$$IGD_{\mathcal{C}} = \sum_{\mathbf{q}\in\chi_{unk}(s)} idg_{\mathbf{q}}(s) \tag{3.47}$$

where

$$igd_{\mathbf{q}}(s) = \lim_{vol(\mathcal{V}(s)\to 0)} \frac{-E\{\triangle H(Q)\}}{\frac{s}{vol(\mathcal{V}(s))}} \tag{3.48}$$

describes the IGD components for the individual configurations. This is the basis for the noisy beam strategy, specified – among other planning strategies – in the next section.

3.6 Planning Strategies

In this section, the geometric models specified in Sec. 3.2.3 are applied to the view planning methods described in Sec. 3.2.2, resulting in comprehensive planning strategies.

The $\mathcal{C}$-space entropy, $H(\mathcal{C})$, is given by:

$$H(\mathcal{C}) = -\sum_{Q_1=0,1} ... \sum_{Q_N=0,1} P[Q_1, ..., Q_n] \log P[Q_1, ..., Q_N]$$

Q_i denotes the random variable corresponding to a configuration q_i being free (=0) or in collision (=1), while $N \in \mathbb{Z}+$ is the number of robot configurations in C-space, discretized with an appropriate resolution. $P[]$ denotes the probability of the corresponding outcome. This allows the computation of the expected IG (or entropy reduction) $\triangle H(\mathcal{C})$ after sensing a region $\mathcal{V}(s) \in \mathcal{P}$ (obstacle/free). Here, s is used to denote a sensor configuration that completely determines a sensing action; $\mathcal{V}(s)$

is the region within the sensor FoV at s. The expected IG is formulated as

$$IG_{\mathcal{C}}(s) = -E\{\underset{s}{\triangle} H(\mathcal{C})\}$$

where E denotes the expectation operation (see Sec. A.1).

In order to implement a sensor-based planning algorithm, a sensor model is required. The basic sensor model geometries employed are detailed in Sec. 3.2.3.

The point model assumes the sensor to have a point FoV. This typically means that the point expected to yield the highest IG will be selected. The beam model, on the other hand, has a straight line FoV considering occlusion by known obstacles. The generalized model is a set of beams.

In the following, the computation of the NBV is described. In general, the IG assumes a Poisson Point Process [149] for obstacles in the physical space and ignores mutual information terms, i.e. joint angles $q_1..q_{DoF}$ are considered independent, because the influence of mutual information is negligible [176], thus, its computation is not considered in this thesis. The following methods apply the MER criterion, either used idealized or noisy.

3.6.1 Ideal Point Method

A detailed derivation of the point sensor for MER is described in [177]. The point sensor has a zero volume FoV. For this reason, the information gain density (IGD) is employed, approximated by $\widetilde{IGD}$ for the sampled case as follows:

$$\widetilde{IGD}_{\mathcal{C}} = \sum_{\mathbf{q}\in\chi_{unk}(s)} igd_{\mathbf{q}}(s) = \sum_{\mathbf{q}\in\mathcal{X}_{unk}(s)} -\lambda \cdot \log(1 - p(\mathbf{q})) \tag{3.49}$$

where $igd_{\mathbf{q}}(s)$ describes the IGD for a configuration $\mathbf{q}$; $\mathcal{V}(s)$ denotes the sensing FoV; and $\mathcal{V}_{unk}(s)$ is the unknown part of $\mathcal{V}(s)$ located in front of the first known obstacle in sensing direction. $\chi_{unk}(s)$, the unknown *C-zone* of the sensing beam s, is defined as the set of configurations whose collision status is unknown and at which the robot intersects with $\mathcal{V}_{unk}(s)$. $p(\mathbf{q})$ denotes the probability that a configuration $\mathbf{q}$ is collision-free, and is given by

$$p(\mathbf{q}) = e^{-\lambda \cdot vol(\mathcal{A}_{unk}(\mathbf{q}))}, \tag{3.50}$$

where $\mathcal{A}_{unk}(\mathbf{q})$ is the unknown part of $\mathcal{A}(\mathbf{q})$.

3.6.2 Ideal Beam Method

In [177], explicit closed-form expressions are derived for $IGD_{\mathcal{C}}(s)$ of an ideal beam sensor, assuming a Poisson Point Process for obstacles in

the physical space. A beam sensor is characterized by a sensing ray of length $L = d_{max}$ starting at the sensor origin (no minimal sensing distance d_{min} is considered). It describes the distance to the closest obstacle point (hit-point) in beam direction.

The beam sensor model has a zero volume FoV, therefore the IGD [177, 188] rather than IG is computed. The final expression of $\widetilde{IGD}$, i.e. the approximation of IGD for the ideal beam sensor, is given by

$$\widetilde{IGD}_{\mathcal{C}} = \sum_{\mathbf{q}\in\chi_{unk}(s)} igd_{\mathbf{q}}(s) = \frac{-\lambda}{L} \cdot \sum_{\mathbf{q}\in\mathcal{X}_{unk}(s)} len(\mathcal{A}(\mathbf{q}) \cap \mathcal{V}_{unk}(s)) \cdot \log(1 - p(\mathbf{q})) \ . \tag{3.51}$$

3.6.3 Noisy Beam Method

The derivation of the C-space entropy computation applying the sensor uncertainty model to a beam sensor is described in this section. For the complete derivation, the reader is referred to [159]. The goal is to plan where to look (sense) next. This entails the determination of an expected outcome of a potential sensor reading based on certain probabilistic assumptions about the environment (e.g., Poisson Point Process for obstacle distribution), an inverse computation compared to building a world map [46].

Noisy sensory data is interpreted as a blurring process, where an ideal sensing P^{δ}_{world} is processed by a noisy system, i.e. a convolution of the ideal sensor result with a Gaussian distribution. The mean of this is the position of the actual obstacle. Therefore, given the condition that an obstacle is at position d_{obs}, the probability of the world model is given by the following equation, ^{G} being used to denote a sensor with Gaussian noise and $D = x$ representing the sensing range:

$$\begin{aligned} &P^{G}_{world}(D = x|d_{obs}) \\ =&P^{\delta}_{world}(D = x|d_{obs}) * N(0, \sigma_d(d_{obs})) \ . \end{aligned}$$

If the entire beam is free, the conditional probability of the world model is given by

$$\begin{aligned} &P^{G}_{world}(D = x|\neg d_{obs}) \\ =&P^{\delta}_{world}(D = x|\neg d_{obs}) * N(0, \sigma_d(d_{max})) \ . \end{aligned}$$

Nevertheless, a probability distribution assumption of the obstacles in the robotic environment is required, using the Poisson Point Model. For view planning, the interest is on the inverse computation, or the conditional probability of having an obstacle at position x_{obs}:

$$P(d_{obs}|world) = \int_{d_{min}}^{d_{max}} P(d_{obs}) P^{G}_{world}(D = x|d_{obs}) dx \tag{3.52}$$

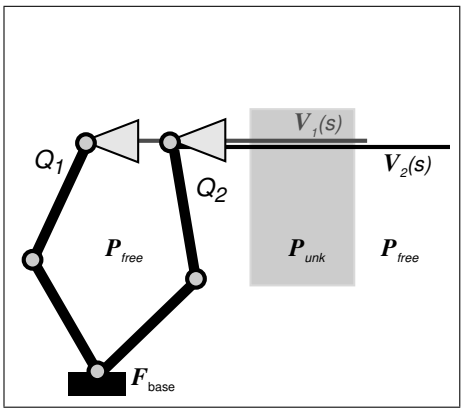


Figure 3.9: Configurations Q_1 and Q_2 will yield the same IG under ideal sensing conditions. Q_2 will be chosen as it results in a higher sensing accuracy (closer to $\mathcal{P}_{unk}$).

In the above equation, $P(x_{obs})$ is used to denote the obstacle probability given by the Poisson Point Process. Similarly, the conditional probability that the beam is free is described as

$$P(\neg d_{obs}|world) = \int_{d_{min}}^{d_{max}} P(\neg d_{obs}) P^{G}_{world}(D = x|\neg d_{obs})dx \tag{3.53}$$

The ideal beam method [177] subdivides the expected entropy reduction (i.e. IG) computation into three events:

$$\underset{s}{E_{beam}\{\triangle H(Q)\}} = \underset{x_{obs}\in\mathcal{A}_{unk}(\mathbf{q})}{E\,\{\triangle H(Q)\}} + \underset{x_{obs}\notin\mathcal{A}_{unk}(\mathbf{q})}{E\,\{\triangle H(Q)\}} + \underset{\exists no\ hitpt}{E\{\triangle H(Q)\}} \tag{3.54}$$

The expected entropy reduction for the blurred beam sensor equivalently consists of a subdivision of events, which are used to compute the IG using the formulae developed above.

The following three events must be considered:

1. there is a hit-point inside $\mathcal{A}_{unk}(\mathbf{q}) \cap \mathcal{V}(s)$;
2. there is a hit-point outside $\mathcal{A}_{unk}(\mathbf{q}) \cap \mathcal{V}(s)$;
3. there is no hit-point within the measurement range.

This leads to the following equations that formulate the expected values of entropy reduction in different cases, as well as the corresponding IGDs:

$$\begin{aligned} \underset{s}{E_G\{\triangle H(Q)\}} &= \underset{x_{obs}\in\mathcal{A}_{unk}(\mathbf{q})\cap\mathcal{V}(s)}{E\ \{\triangle H(Q)\}} + \underset{x_{obs}\notin\mathcal{A}_{unk}(\mathbf{q})\cap\mathcal{V}(s)}{E\ \{\triangle H(Q)\}} + \underset{\exists no\ obs}{E\{\triangle H(Q)\}} \\ &\Leftrightarrow (igd_q) = (igd_q)_1 + (igd_q)_2 + (igd_q)_3 \end{aligned} \tag{3.55}$$

The first event corresponds to a hit-point being inside $\mathcal{A}_{unk}(\mathbf{q}) \cap \mathcal{V}(s)$. Thus, $\mathbf{q}$ is in collision after sensing. The IGD is given by:

$$(igd_q)_1 = \frac{\lambda H(Q)}{M_{obs}} \cdot \int\limits_{x \in \mathcal{A}(q) \cap \mathcal{V}_u(s)} \int_{d_{min}}^{d_{max}} P^G_{world}(D = x | x_{obs}) dx_{obs} dx \tag{3.56}$$

where M_{obs} denotes a normalization factor.

In the second event, the hit-point is outside $\mathcal{A}_{unk}(\mathbf{q}) \cap \mathcal{V}(s)$, but inside $\mathcal{V}_{unk}(s)$. The entropy reduction is induced by the newly sensed regions (the part of the $\mathcal{V}_{unk}(s)$ in front of the hit-point). However, the Poisson Point assumption (obstacles are points with no volume) induces that the expectation (even after being blurred by the sensory noise) is zero. The result is given by:

$$(igd_q)_2 = 0 \ . \tag{3.57}$$

In the third event, the entire beam is sensed free. The result is given by:

$$(igd_q)_3 = \frac{1}{M_{free}} \frac{dH(Q)}{dV} \cdot \int\limits_{x \in \mathcal{A}(\mathbf{q}) \cap \mathcal{V}_{unk}(s)} \int_{d_{min}}^{d_{max}} P^G_{world}(D = x | \neg x_{obs}) dx_{obs} dx \tag{3.58}$$

where M_{free} is used for normalization. The factors

$$M_{obs} = \int_0^{d_{max}+2 \cdot \sigma_{max}} \int_{d_{min}}^{d_{max}} (p(D = x | x_{obs}) + p(D = x | \neg x_{obs})) dx_{obs} dx \tag{3.59}$$

and

$$M_{free} = \int_0^{d_{max}-2 \cdot \sigma_{max}} p(D = x | \neg x_{obs}) dx \tag{3.60}$$

are the normalization factors. The derivatives of the entropy $H(Q)$ w.r.t the volume lead to

$$\frac{dH}{dV} = \frac{dH}{dp} \frac{dp}{dV} = -\frac{dH}{dp} \lambda p(\mathbf{q}) \tag{3.61}$$

$$\frac{dH(Q)}{dp} = log(p(q)) - log(1 - p(\mathbf{q})) \tag{3.62}$$

For computational reasons, the formulations are discretized, e.g. $x(n) = x(t = nT)$ with $T = \frac{\sigma_{min}}{2}$, and stored in Look-up tables.

Eq. (3.55) enables the computation of the IGD for each sensing action s. The IGD depends on the sensing inaccuracy, i.e. variance σ_d. The accuracy is higher for smaller ranges. Therefore, the IGD will be low for sensings reaching farther away from the sensor origin, as shown in Fig. 3.9. The view planning result is a trade-off between sensing accuracy and accessibility.

3.6.4 Generalized Method

In [179], a closed-form expression for the MER criterion based on a generalized non-zero volume FoV sensor model is presented. Most sensors commercially available (cf. Fig. 2.5) provide range images, either 2-D for laser-range scanners or 2.5-D for stereo sensor. The generalized method is an extension of the beam method, it considers n-beams, as depicted in Fig. 3.3. The drawback of the higher computational complexity is compensated by improved exploration results (cf. [180]). As the volume of the FoV is not zero, the IG rather than the IGD is considered, leading to the following equation:

$$\widetilde{IG}_{\mathcal{C}} = -E[\Delta H(\mathcal{C})] = \sum_{q \in \chi_u(s)} ig_{\mathbf{q}}(s) \ . \tag{3.63}$$

A detailed derivation of the method is presented in [179]. The application of the method to noisy computation, as derived for the beam method, is not recommendable due to the complexity of the method. Therefore, the following method based on adaptive obstacle density is proposed.

3.6.5 Adaptive Obstacle Density Method

The map of the environment reflects the occupancy state, i.e. the probability for the existence of an obstacle. In case of ideal tristate representation (occupied, free, unknown), it is impossible to derive the information *nearly free* or *nearly occupied*, i.e. information on the obstacle density is not obtainable. Therefore, using a constant density for the Poisson Point Process is reasonable. Previous results (cf. [180]) show that the choice of density λ does not influence the exploration results when kept constant during exploration. The intensity value does not represent the true state of the environment, therefore λ can be selected by chance.

The use of an occupancy grid extends the knowledge of the physical space. By interpreting the continuous state of the cell as intensity, a inhomogeneous Poisson Point Process is used.

In the following, all cells of $\mathcal{P}$-map are assumed to be independent. Each cell c_i can be conceived as a separate process with an individual density λ_i, making the ideal beam and ideal point sensor directly applicable to this environment by adapting λ with λ_i. Initially, all $\lambda_i = 0.5$); the more measurements are performed, the more adaptively λ_i guides the exploration. The void probability, i.e. the probability of a region being free of obstacles, is presented for the marginal cases $\lambda_i \to 0$ and $\lambda_i \to 1$ in the following:

- If $\lambda_i \to 0$, the cell is assumed to be free. The corresponding void probability, assuming Poisson Point Model, is

$$\lim_{\lambda_i \to 0} (p(c_i) = e^{-\lambda_i |c_i|}) = 1 \ .$$

  This result shows that the void probability is now influenced by the volume and the current state of the cell. The larger and emptier the cell, the higher the void probability.

- The more occluded the cell, the lower the void probability, i.e.

$$\lim_{\lambda_i \to 1} (p(c_i) = e^{-\lambda_i |c_i|}) = e^{-|c_i|}$$

  . Therefore, the entropy will be maximally reduced. This guides the robot to explore occupied regions with high priority.

The grey values for the cells are computed for the individual updates as follows:

Applying Bayes' Rule, the state of a cell is computed using the respective Log-Odd l_{occ}

$$p(c_i) = \frac{1}{1 + l_{occ}} \ . \tag{3.64}$$

When using Belief Update for incorporating uncertain sensings, the state of the grid is computed by combination of the belief in $m_{c_i}(free)$ and $m_{c_i}(occ)$ for each respective cell as follows:

$$p(c_i) = \frac{m_{c_i}(occ) - m_{c_i}(free) + 1}{2} \tag{3.65}$$

When applying Fuzzy Update, the state for each cell is computed identically:

$$p(c_i) = \frac{m_{c_i}(occ) - m_{c_i}(free) + 1}{2} \tag{3.66}$$

Each update allows the use of the grey value as an indication of the intensity for the Poisson Point Process without violating the probability theory. This statement even remains valid if Fuzzy Update is applied.

The states of the occupancy grid $p(c_i) \in [0, 1]$ are used to estimate an intensity value for the inhomogeneous Poisson Point Process, which usually covers $\lambda \in]0, \infty[$; the intensity $\hat{\lambda_i}$ is estimated

$$\hat{\lambda_i} = \frac{p(c_i)}{1 - p(c_i)}; \ p(c_i) \in]0, 1[\ . \tag{3.67}$$

The Adaptive Obstacle Density Method (AOD) allows the use of existing planning methods while adapting them to the use of occupancy maps. The variable λ is the intensity inhomogeneous Poisson Point Process. The parametrization based on the cell's state is a simple but realistic estimate. The application of estimators known in spatial statistics is

not possible, especially due to the incremental process. The occupancy grid's states, i.e. grey scale values, provide a measure for λ. The only reasonable way to estimate λ is the use of a transformation of these states. The update rule used to estimate the state plays an inferior role in the process, as the interpretation of the grey scale values stays the same. The computation of the scale grey values is performed ad hoc, therefore it remains a matter of discussion.

The beauty of simplicity allows an application to the ideal planning methods. The view planning theory, i.e. the computation of the $\mathcal{C}$-space entropy, remains unchanged, still a more realistic environment model is considered in the planning stage. The variation of λ is also applicable to the noisy beam method. The increase of the computational effort can be neglected as opposed to the noisy beam approach.

3.7 Sampling of Work Space

The performance of the $\mathcal{C}$-space exploration not only depends on view planning strategy and sensor model, but also highly on the sampling strategy used for generating configurations for view planning. Fig. 3.10 shows an exemplary $\mathcal{C}$-space, which is referred to throughout this section. In general, there are two ways of generating a suitable image of $\mathcal{C}$:

1. computation of an algebraic solution to generate an exact representation of $\mathcal{C}_{free}$, i.e. sets F and G in Fig. 3.10;

2. computation of a probabilistic solution generating a suitable subset of $\mathcal{C}_{free}$, i.e. $\mathcal{R}_1$ and $\mathcal{R}_2$ in Fig. 3.10.

While 1. requires a prohibitive amount of time to be computed for high-dimensional $\mathcal{C}$-spaces in non-trivial environments, exact fast algorithms exist for 2. to test whether a configuration $\mathbf{q}$ or path $\mathcal{L}$ is collision-free, i.e. $\mathcal{L}(\mathbf{q}_2, \mathbf{q}_1) \in \mathcal{C}_{free}$. Details on collision detection algorithms can be obtained from [100].

For exploration, a representative set of accessible (connected) free nodes, i.e. possible view nodes, is aspired. Furthermore, unknown nodes, i.e. explore nodes, must be computed. Both subsets represent the set $\mathcal{C}_{free}$ and $\mathcal{C}_{unk}$ respectively. The two sets cannot be explicitly computed in high-dimensional configuration spaces, thus view and explore nodes are obtained by sampling. In Fig. 3.10, a typical sampling of $\mathcal{C}$-space is visualized. As both sets are sampled, the sampling technique has a great impact on the exploration results.

Probabilistic Roadmaps (PRM) apply these algorithms in order to check whether

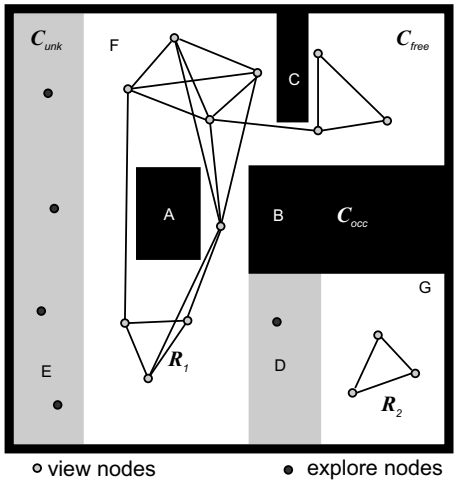


Figure 3.10: $\mathcal{C}$-space representation: the separate sets of $\mathcal{C}$ are denoted A to G. The view nodes are connected linearly in a roadmap obtained by unbiased sampling with subsets R_1 and R_2.

- any sampled $\mathbf{q} \in \mathcal{C}$ is inserted in $\mathcal{R}$ iff $\mathbf{q} \in \mathcal{C}_{free}$, where $\mathcal{R}$ is called a roadmap of free configurations;
- any two configurations $\mathbf{q}_1, \mathbf{q}_2 \in \mathcal{C}$ are connected by a straight line $\mathcal{L}$, representing edges in $\mathcal{R}$.

PRMs never compute the exact shape of $\mathcal{C}_{free}$, they provide the connectivity of the nodes, e.g. the set R_1 does not provide information on the shape of A. The efficiency of the PRM in an exploration application massively depends on the sampling method.

One major factor influencing the performance of PRMs [76] is the sampling measure, e.g. uniform sampling, in contrast to a negligible influence of the sampling source, e.g. random numbers with constant seed. Usually, uniform sampling is outperformed by non-uniform sampling, such as connectivity expansion or Gaussian bias.

In this thesis, PRMs for sensor-based exploration are considered. The key challenge is to generate nodes with large insight on unknown areas of $\mathcal{C}$. In order to achieve a good visibility, biases can be assigned.

A configuration $\mathbf{q}_{sample} \in \mathcal{C}_{free}$ is sampled and is connectable to the roadmap. Before insertion of $\mathbf{q}_{sample}$ into the roadmap, the distance between $\mathbf{q}_{sample}$ and the unknown configurations $\sum \mathbf{q}_{unk}$, which is also obtained by sampling the unknown areas of $\mathcal{C}$, is computed. Then, the unknown configuration $\mathbf{q}_{unk}$ which is closest to $\mathbf{q}_{sample}$ is selected. Lastly, $\hat{\mathbf{q}}_{sample}$, which is obtained by sampling $\mathcal{L}(\mathbf{q}_{sample}, \mathbf{q}_{unk})$ is inserted, with $\hat{\mathbf{q}}_{sample}$ being as close as possible to the border of $\mathcal{C}_{unk}$. In literature, this approach is often called *sampling with $\mathcal{C}$-space boundary constraint* [91]. The method enables optimal motion planning in narrow passages.

The $\mathcal{C}$-space boundary bias provides good roadmaps for motion planning. Sensing is performed in physical space, therefore the set of view nodes requires a sampling method with a $\mathcal{P}$-boundary constraint. This technique is particularly well-suited for sampling $\mathcal{C}_{free}$ in order to obtain nodes for view planning, constrained by $\mathcal{V} \cap \mathcal{C}_{unk} \neq \emptyset$. Results from different sampling techniques are presented in Chap. 6.

3.8 Conclusion

This chapter presents the various methods for work space exploration, a major prerequisite when acting in a partially unknown environment with complex robots. The focus is on computing an information measure for efficient exploration of the maneuverability of the robot, denoted $\mathcal{C}$-space entropy. The views are planned by maximizing IG, i.e. the expected reduction of entropy of $\mathcal{C}$-space. Measurements are performed in physical space and mapped from $\mathcal{P}$ to $\mathcal{C}$ by the robot kinematics and geometry.

Sensing in real world applications is prone to impreciseness and errors on non-perfect surfaces. Therefore, the methods for integrating uncertain sensings into a consistent grid-based map of the environment based on three different update types (Bayes, Belief, and Fuzzy) are introduced. Bayes and Belief require independency of the sensings, whereas Fuzzy requires more research in formulation of the membership functions.

The uncertainty in physical space must be considered in computing the views for exploration of $\mathcal{C}$-space. Two main methods are developed in this section:

- Noisy Beam Method (NB);
- Planning based on Adaptive Obstacle Density (AOD).

While the first requires sophisticated computations only feasible for the beam planning method, the second method is applicable to all planning methods, i.e. point, beam, and generalized. The state of the physical space map is used to adapt the density of the Poisson Point Process, applicable independently of the update rule applied. This transition from a homogeneous to an inhomogeneous Poisson Point Process enables the consideration of uncertainty in the planning stage.

The adaption of the intensity λ based on the state of a cell is reasonable, as λ is no probability, therefore the above constraint is not relevant and the method is applicable to all update types. The naïve update type relates to the idealized planning methods and is applied to emphasize the necessity of measurement interpretation.

Different sampling methods for navigation and view planning are described. It is proposed to sample view nodes and exploration nodes to compute the entropy. The influence of the sampling method on the exploration process is assessed.

The methods developed in this section are evaluated in the simulations in Chap. 6 in order to derive proper criteria for the heuristics, e.g. sampling with respect to the planning strategy, and sensor characteristics required for efficient exploration.

4

Exploration of Physical Space

In this chapter, the task-directed exploration of physical space, i.e. the computation of views acquiring objects represented in $\mathcal{P}$-space, is described in detail.

The previous chapter focuses on methods for gaining information on the $\mathcal{C}$-space by defining $\mathcal{C}$-space entropy as a measure to compute greedily optimal sensor poses. The $\mathcal{C}$-entropy is mapped to $\mathcal{P}$-space using the robot's non-trivial geometry and kinematics, leading to an information gain (density) map $\mathcal{I}$. The work space, i.e. the space accessible by the robot, is a subset of the physical space $\mathcal{P}$. In this section, the exploration task is extended towards exploring $\mathcal{P}$-space. The exploration method is elaborated for the automatic modeling task, however, it can be extended to other missions. The approach merely requires minimal a priori knowledge, i.e. the approximate pose of the object to be modeled.

This chapter is organized as follows: The problem of exploring of $\mathcal{P}$-space is introduced. Existing automatic inspection algorithms are revisited and their suitability for articulated systems with many DoFs is elaborated. Then, a sampling-based method for exploring objects based on the definition of a regions of interest (RoIs) is developed. The computation of the IG and the limitation of the search space for possible views in $\mathcal{P}$-space exploration is presented. Conclusive remarks on the derived NBV approach close the chapter.

4.1 Problem Statement

In most current approaches, a model of the work object is given and views are planned on this basis. In this thesis, a model-free approach is used: When exploring objects in partly unknown environments, it is hard to justify the large amount of a priori knowledge that would be required for model-based approaches. In the following, the necessity of limiting the search space for view planning is elaborated. Then, the general problem of model generation is addressed.

4.1.1 General Approach

In a first step, at least one Region of Interest (RoI) is defined either manually or automatic. This RoI provides an upper bound of the parameter space for view planning. In the next step, a sensing and classification strategy to detect RoIs (e.g. by using stereo vision combined with segmentation and classification routines) as target areas for an inspection of the unknown environment is developed. Two approaches for RoI specification are elaborated: the RoI is defined by a sphere with radius r_{RoI} and center $\mathbf{c}_{RoI}$; the RoI is modeled as a cuboid, its lower left corner $\mathbf{x}_{RoI}$ and size $\mathbf{s}_{RoI}$ define its location and dimension.

In case the RoI is modeled as a sphere, i.e. view sphere, the object to be inspected is situated completely inside this sphere, thus all viewing poses for the NBV algorithm will be situated on or, more literally, in this sphere. The gross RoI information being preliminary w.r.t. accuracy, r_{RoI} can be modified during the inspection task, which allows for the sensor to be placed inside the sphere. The radius r_{RoI} should be an upper bound for the area occupied by the objects of interest.

The representation of the RoI as a cuboid allows the sensor to enter the RoI as knowledge becomes available. In this case, the RoI does not discretize the space of possible views on a sphere, but merely limits the location of the object.

In the next section, a short summary of the modeling approaches is given.

4.1.2 3-D Model Generation

The task of most inspection systems is to generate model representations of objects. Usually, volume-based and surface-based representations are distinguished. Additionally, most approaches do not allow for an incremental generation of the model, as the assumption that all data is known before the model reconstruction process prevails in computer graphics.

Based on previous work from [40] and [168], large data sets have been computed by [95] during the Digital Michelangelo Project (cf. Sec. 2.3.2). The views, i.e. different scans, were chosen manually and minor manual alignment of the individual scans was applied.

Work performed at Columbia University in New York [7, 8, 9] generates models of historic sites: Mobile robots are used for data acquisition, a coarse volume-based representation guides the system. Further research deals with model construction from range images [146], an optimization of surface meshes is proposed in [42].

Slabaugh et al. [142] present methods for 3-D volumetric reconstruction of visual scenes, photographed by multiple calibrated cameras placed at arbitrary view points. Aiming at a 3-D model that can be rendered to synthesize photo-realistic views, they improve existing voxel coloring and space carving approaches.

In this thesis, the incremental method described in [19] is applied for surface model generation. Volume-based modeling is used for the complete work cell, implementing a space carving approach.

4.2 Methods for Next-Best-View Computation

In this section, a short overview on research in the field of automatic model generation is presented, focusing on the view planning aspect. The main differences between today's common approaches is the data basis:

- views are planned based on ideal models of the object;
- no assumptions on the nature of the object are made.

When handling volumetric data, the problem of view point selection is significantly larger than in 2-D. The planning of promising sensor poses is a variant of the art gallery problem (cf. Sec. 2.1.6). In the original problem, sensors have unlimited vision in range and 360 degrees opening angle, surfaces can be seen from any angle. In reality, this is never the case: Sensors have limited vision, usually a minimal and maximal sensing range, and the grazing angle, i.e. the maximal angle under which a surface can still be measured, depends on the sensing principle (time-of-flight, triangulation) as well as reflectivity (specular, diffuse) and color (texture) of the surface. This section focuses on planning NBVs in physical space, sensing objects or complete environments.

Existing methods are differentiated by the strategy used as a guide for planning views, i.e. silhouette, occlusion, entropy; further approaches complete the discussion. Based on this discussion, the selection of the suitable method for the approach performed in this thesis is elaborated.

4.2.1 Silhouette-Based Methods

The work of Stuerzlinger [152] focuses on view point selection for image-based modeling by sampling all visible surfaces; however, the coverage is not addressed. In order to reconstruct the viewable surface of an object completely, multiple views of the same object must be integrated into a common coordinate system.

Fitzgibbon et al. [50] describe a system which, given a sequence of images of an object rotating about a single axis, generates a textured 3-D model fully automatically. This technique neither requires prior information about the cameras or scene, nor any knowledge of the turn-table angles.

4.2.2 Occlusion-Based Methods

Without model data being available, a common approach is to use occlusion as a guide to the NBV. The work of Maver et al. [108] is extended to more DoFs by Pito [124, 125]. Pito defines a discretized positional space to assign an upper bound to a set of view poses. Occlusion is projected onto a positional space to determine view poses looking behind already acquired surfaces. Tarabanis et al. [160] consider self-occlusion for guiding the inspection process: given a model feature to be investigated, valid view points free of occlusion are computed. Reed et al. [127] use a surface-based representation of the object, and sweep the surface in order to extrude a solid model of both the imaged surface and the occluded volume. The surfaces are swept in viewing direction and fused to a single model using a set operator. The swept volume approach of Abrams et al. [2, 3] permits arbitrary polyhedral objects (given in a typical boundary representation) to be swept through an arbitrary trajectory.

When inspecting mesh-based objects with significant self-occlusion, the common approach is to use histograms. These contain the surface normals on the volume of occlusion, weighted by area, picking the maximum value to generate the NBV.

In Fig. 4.1, the principal modeling process is described. A sensor with limited sensing range and limited opening angle senses a surface. This is done exemplary for a cylindrical range sensor. Each hit-point on the surface creates an occluded patch, its size being determined by the sensing range, the maximal sensing range, and the angular resolution.

The following computations refer to Fig. 4.1: The sensor measures distances $d \in [d_{min}, d_{max}]$. It has an angular resolution $\delta\alpha$, the opening angle is defined by $\triangle\alpha = \alpha_{off} - \alpha_{on} = \sum_{i=1..k_d} \delta\alpha_i$. The number of individual measurements per scan is denoted k_d. The main task is to compute the occluded volume per scan as shown in the lower plot in Fig. 4.1. The occluded volume is denoted A_{occl}. This occluded volume

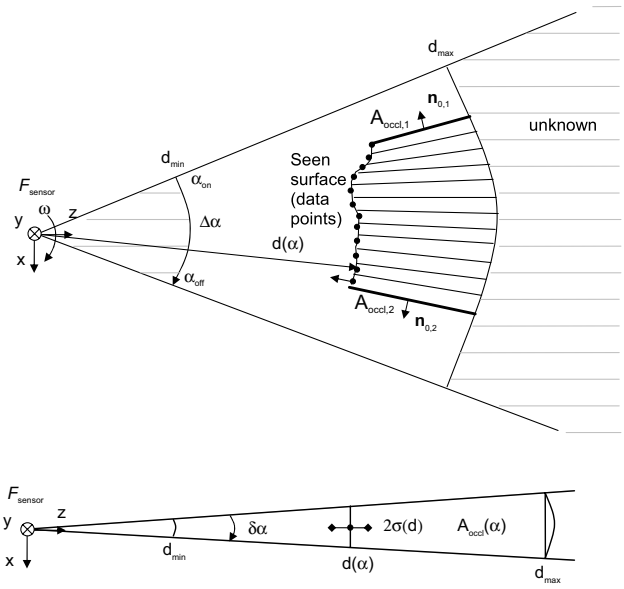


Figure 4.1: Occlusion-based NBV calculation for a cylindrical range sensor: The sensor performs k_d independent measurements per scan. The vectors $\mathbf{n_{o,1}}$ and $\mathbf{n_{o,2}}$ describe the orientation of the occluded volume for each scan.

is then projected to the view sphere as shown in Fig. 4.2.

In each measurement, the computation of the occluded volume is performed as follows (cf. Fig. 4.1):

$$A_{occl}(\alpha, d) = 2 \cdot \tan(\frac{\delta\alpha}{2}) \cdot d \cdot (d_{max} - d) + \tan(\frac{\delta\alpha}{2}) \cdot (d_{max} - d)^2 \tag{4.1}$$

The substitution of $\tan(\frac{\delta\alpha}{2}) = k$ leads to:

$$A_{occl}(\alpha, d) = k \cdot (d_{max}^2 - d^2) \tag{4.2}$$

In this thesis, noise is considered in the planning stage. Therefore, a noise factor, i.e. variance σ, is applied. The occluded volume is computed and projected to the outer borders of the obstacle and projected onto the view sphere. In the projection, the grazing angle, i.e. the angle between the orientation of the volume denoted by $\mathbf{n}_{o,1}$ and $\mathbf{n}_{o,2}$ as well as the view vector is considered. The view vector is oriented towards the center of the view sphere. It describes the possible NBV. Based on the occluded volume:

$$|\mathbf{n}_{o,1}| = A_{occl,1} = \sum_{\triangle\alpha} A_{occl}(\alpha, d) \cdot \frac{\alpha_{off} - \delta\alpha}{(\alpha_{off} - \alpha_{on})} \tag{4.3}$$

and

$$|\mathbf{n}_{o,2}| = A_{occl,2} = \sum_{\triangle\alpha} A_{occl}(\alpha, d) \cdot \frac{\delta\alpha}{(\alpha_{off} - \alpha_{on})} \tag{4.4}$$

the norm of the vectors is computed.

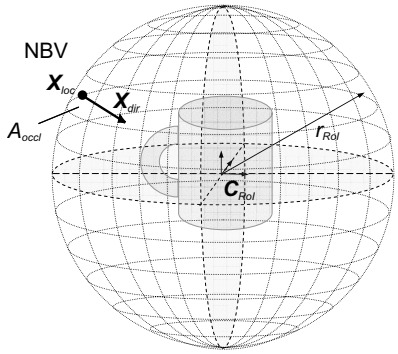


Figure 4.2: Region of Interest, i.e. a discrete positional space, modeled as a view sphere. The view sphere bounds the possible NBVs.

The scalar product between the cell's view direction, i.e. the view vector, and $\mathbf{n_{o,i}}$; $i = 1, 2$ defines the value of the respective cell. A grazing angle can be considered. This mapping process on the view sphere is performed for every edge in every iteration. Then, the maximal grid value on the tesselated view sphere is selected as the NBV pose.

This method is computationally expensive, although the search space for possible view points is limited to a viewing sphere. It is mostly applied in modeling convex objects, and mainly tested on turn-tables and sensors with unlimited vision (in relation to the inspected object's dimensions).

The approach of selecting a view sphere to limit the view space is suitable for the sensor system used in [125], i.e. a turn-table. In this thesis, the RoI merely coarsely defines the object to be modeled, as the exact location is not precisely known. A robot with complex kinematics, which is able to enter the objects bounding box, is applied, thus favoring the IG-based method assessed in Sec. 4.2.5.

4.2.3 Entropy-Based Methods

A recent series of publications of Chen et al. [35, 36, 96] deals with automated model-based inspection, applying an entropy measure to guide the inspection. Wenhardt et al. [183] present an entropy- and model-based algorithm for optimal view point selection for 3-D reconstruction of an object, using 2-D image points. Due to noisy image points, they apply a Kalman filter to obtain the best estimate of the object's geometry. This filter allows an efficient prediction of the effect any given camera position has on the uncertainty, and therefore quality, of the estimation.

Liska et al. [103] address the problem of a varying resolution in camera direction. The camera system is attached to a turn-table in order to perform automatic modeling. The authors derive a relation between

entropy and the next viewing angle in order to tackle this problem. Vazquez et al. [173] present the view point entropy, extending their earlier work [172] to computing views of large molecules. The focus is set on an algorithm implemented on the graphics board to compute good views using a 2-D screen.

4.2.4 Further Methods

Whaite and Ferrie [185] assume that a scanned object can be approximated by a collection of parts modeled with superquadrics. They extend this approach [186] to a system which uses uncertainty measures based on superquadrics to drive an exploration process.

The work of Banta et al. [12] computes view points on ideal work objects, i.e. a model-based approach; recently work in this direction is also performed by Laurentini [90] and Bottino [22].

Agathos et al. [5] fuse images and range, i.e. texture mapping, in order to guide a modeling process.

Blaer et al. [17, 18] present an approach to modeling urban environments, similar to the one presented in this thesis, yet at much larger scale. A coarse volume-based model is derived from aerial data of an urban scene. Based on this model, robot poses are planned in order to gain a precise surface-based model using a mobile robot and a laser-range scanner. The view planning space for the poses is 3-D, as the authors use a mobile robot moving in a plane, i.e. $\mathrm{x} = [x, y, \theta]$. The high dimensionality of the view planning space applied in this thesis makes this approach intractable, nevertheless, the coarse planning of views based on a voxel-based representation in order to generate a surface mesh of higher resolution is a similarity.

Gonzalez et al. [59] show a sampling-based approach solving the art gallery problem in 2-D in a polygonal work space, considering the grazing angle for planning views. In case no depth measurement is obtained from the sensor, the authors compute the boundary of $\mathcal{P}_{free}$ in a polygon-based environment depending solely on the grazing angle in order to estimate the unseen surfaces.

4.2.5 Assessment of the Suitable Method

Most methods for object modeling require CAD models in advance, based on which views for the real sensor are computed. In this thesis, these approaches are not applicable, as the robot system is to explore a partly unknown physical space. Computing views based on known CAD-models is merely necessary for object registration, i.e. sensing the object to estimate its pose. In this case, the object's pose is not exactly known, therefore an approximate region must be explored.

The works of Gonzalez et al. [59] and Blaer et al. [17, 18] encourage the approach taken in this thesis.

The view sphere approach detailed in Sec. 4.2.2 is computationally expensive. Additionally, the views are restricted to the view sphere, which does not allow the modeling of objects with high self-occlusion. This method is optimized for turn-tables, the view sphere is transformed into a view cylinder. The method assumes that the sensor system moves on the view sphere. In turn-table settings, the sensor usually cannot physically enter the view sphere, justifying the approach. When modeling arbitrary, i.e. non-convex, objects, this strategy constrains the motion of the sensor too much.

As discussed in Chap. 3, optimal views regarding $\mathcal{C}$-space knowledge are planned based on IG with abstract sensor models. In order to follow this approach, a strategy which limits the search space for the possible views, i.e. the definition of a RoI, is required. The planning strategy must not constrain the motion too rigidly, otherwise the accessability of a large number of view points (which a complex robot offers) is limited unnecessarily, especially in obstructed environments with many occupied regions.

Additionally, it is beneficial if the robot can enter the RoI. Thus, a a more coarse definition of the RoI is sufficient. Furthermore, the planning methods developed in Chap. 3 can be directly applied to the modeling task, and the modeling process can be decoupled from the planning process: Planning is performed in a grid-based representation, while modeling can be performed voxel- and/or surface-based. Occlusion is implicitly considered in the planning model, i.e. beam model or generic model. The resolution of the 3-D modeling and the planning is also decoupled. Additionally, the *safe-for-motion* constraint must be considered, as the robot might collide with the object to inspect.

In this thesis, views are planned based on an entropy measure, allowing a combination with the $\mathcal{C}$-space exploration methods described in the previous chapter. The thesis follows a similar rationale as presented in the recent work of Blaer, extended w.r.t. the entropy measure with Gonzalez' justification for sampling-based approaches in 3-D environments.

4.3 Exploration of Regions of Interest in Physical Space

The approach for view planning elaborated in this thesis fuses sampling techniques for generating views with a voxel-based entropy representation in physical space. Additionally, the approach enables a simultaneous incremental construction of a surface model of independent granularity.

In the following section, the necessity for bounding the search space in $\mathcal{P}$ is elaborated and a method for RoI identification based on range or image data is selected; followed by a description of the means applied for selecting RoIs.

Further, approaches to efficiently compute optimal view points (NBVs), assuring a maximum IG, are detailed. In order to combine the maneuverability exploration with physical space exploration, an entropy-based measure is chosen. Finally, the sampling of view points is elaborated.

4.3.1 Definition of Regions of Interest

The search space for possible view poses must be bounded, otherwise the art gallery problem is computationally prohibitive in 3-D. RoIs can be defined in two ways (cf. Fig. 4.2):

1. The RoI is defined by a sphere, which cannot be entered. Views are planned on the boundary.

2. The RoI is enclosed by a cuboid, allowing the sensor to enter the boundary.

The first definition is more suitable for limiting possible view locations in occlusion-based modeling. The second one is followed in this thesis, as it allows for a combination with the $\mathcal{C}$-space exploration described in Chap. 3. Computation of entropy for guiding the exploration is possible for both definitions.

4.3.2 Selection of Regions of Interest

In completely autonomous operations, a measure for finding suitable RoIs is required. In the scope of this thesis, the selection of RoIs is not performed in the simulations and experiments, as they are manually defined a priori. However, proposal on selecting areas are provided for future implementations.

When the task is to model an object, a RoI can be identified as a region of high structure, i.e. featuring edges in all directions. Edge filters are very common to detect edges in certain directions.

The structure tensor[1] provides a good measure for identifying interesting regions. Fundamental work of Foerstner [51] and Harris and

[1]The term tensor refers to a representation of an array of data. The rank of a tensor denotes the number of indices needed to describe it, e.g. a vector is a tensor of rank one, whereas a two-dimensional matrix is a tensor of rank two. The structure tensor is usually derived from images.

Stevens [63] initiate the use of the structure tensor for low-level feature analysis, such as corner detection, edge detection, texture analysis, and optical flow.

The structure tensor represents gradient information, it is composed of the partial derivatives I_x and I_y in the image I.

The structure tensor matrix is computed as follows:

$$S = \begin{bmatrix} {I_x}^2 & I_x I_y \\ I_x I_y & {I_y}^2 \end{bmatrix} \tag{4.5}$$

The eigenvalues and eigenvectors of S are used as features to allow a more precise description of the local gradient characteristics. The first eigenvector is a normal to the gradient edge, while the second is tangentially oriented. The eigenvalues indicate the underlying gradient structure along their associated eigenvector directions. For computing RoIs, the following strategy can be applied:

- one eigenvalue: edge detection, minimal requirement for a RoI;
- two eigenvalues: structure in both directions, area labeled as RoI.

The structure tensor S in Eq. (4.5) is defined for 2-D images. In order to obtain a 3-D RoI, the depth of the feature points must be computed. The distance can be obtained using stereo vision to define distance and dimension. A 3-D variant of the structure tensor depends on geometrical data such as

$$S(x, y, z) = \begin{bmatrix} {I^2}_x & I_x I_y & I_x I_z \\ I_y I_x & {I^2}_y & I_y I_z \\ I_z I_x & I_z I_y & {I^2}_z \end{bmatrix} . \tag{4.6}$$

The latter is used e.g. in medical analysis of Computer Tomography (CT) data or with laser radar data [116] for feature extraction. Such works show that the selection of RoIs based on 2-D/3-D structure tensors is promising, yet an experimental evaluation is beyond the scope of this thesis.

4.3.3 Computation of the Next-Best-View

For $\mathcal{C}$-space exploration, an entropy measure is defined in Chap. 3. The IG in $\mathcal{C}$ is mapped to the physical space via the volume of the robot in the respective configuration $\mathbf{q}$. In this thesis, the inspection task is integrated into an exploration scenario, requiring a minimal amount of a priori knowledge. The entropy states the current information. It does usually not provide information on the result of a sensing process. The IG is computed by defining an expected entropy reduction in $\mathcal{C}$-space. The challenge is to derive an appropriate IG definition for the inspection task defined in physical space. The Poisson Point Model

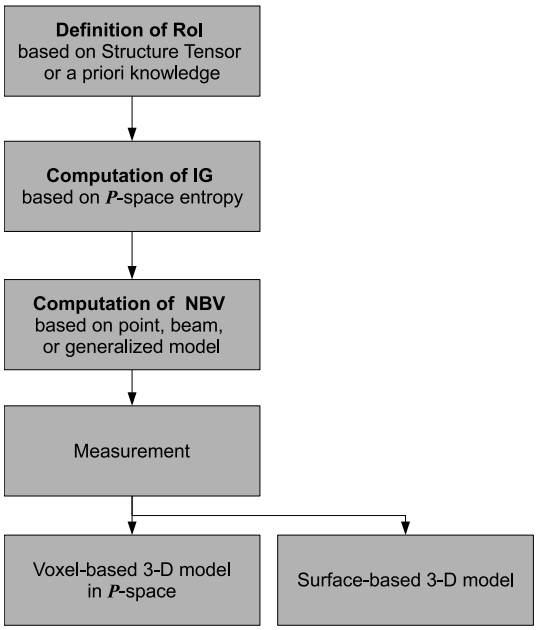


Figure 4.3: Flowchart of the developed view planning methods for model-free object modeling: RoIs are defined a priori. The IG is computed based on $\mathcal{P}$-space entropy (cf. Eq. (4.8)). The measured data is transferred to the surface modeling method [19] and the $\mathcal{P}$-space representation simultaneously.

defines a void probability for a set $\mathcal{B}_{unk}$, e.g. a section of $\mathcal{P}_{unk}$, as follows:

$$p(\mathcal{B}_{unk}) = e^{-\lambda|\mathcal{B}_{unk}|}, \lambda \in]0, \infty[\ . \tag{4.7}$$

In inspection, the goal is to completely model the RoI, i.e. to reduce the number of unknown surface cells to a minimum. The resolution of the cells in a voxel-based representation is crucial for the success of the method. Occlusion dominates inspection, especially as most objects cannot be expected to be convex. Most approaches rely on convexity of the objects to inspect; this constraint is eased by the application of a heuristic for inspection. The focus is on defining RoIs in the physical space and applying a probabilistic measure to guide the inspection.

The entropy is obtained by adapting λ, the intensity of the Poisson Point Process, based on the occupancy state of the respective cell.

$$H(\mathcal{B}_{unk}) = -p(\mathcal{B}_{unk}) \cdot \log p(\mathcal{B}_{unk}) - (1 - p(\mathcal{B}_{unk})) \cdot \log(1 - p(\mathcal{B}_{unk})) \tag{4.8}$$

The adaption of λ can be performed in two ways: On the one hand, λ is directly assigned the grey scale value, i.e. the occupancy state of the cell. The intensity in then limited to the interval $]0, 1]$:

$$\lambda_{c_i} = p(c_i); \lambda \in]0, 1] \ . \tag{4.9}$$

On the other hand, in order to utilize the full range of the intensity, i.e. $\lambda \in]0, \infty[$, the following constraints must be considered: The minimal grey scale value $p_{min}(c_i) = 0$ describes the situation of the cell being

empty. The probability for an obstacle under Poisson Point is zero, the upper boundary for $p_{max}(c_i) = 1$, the cell is known to be occupied. The probability for an obstacle being present is maximal, represented by an intensity $\lambda \to \infty$. If $p(c_i) \in [0, 1]$. The following equations describe the results for $p(\mathcal{B}_{unk})$:

$$\begin{aligned} \lambda \to 0 &: p(\mathcal{B}_{unk}) = 1; \\ \lambda \to \infty &: \lim_{\lambda \to \infty} p(\mathcal{B}_{unk}) = 0 \ . \end{aligned} \tag{4.10}$$

The term for the void probability described in Eq. (4.7) is

$$\lambda_{c_i} = \frac{p(c_i)}{1 - p(c_i)}; \ \lambda \in]0, \infty[, p(c_i) \in [0, 1] \ . \tag{4.11}$$

The variation of the intensity is applicable to all update models (Bayes, Belief, and Fuzzy) described in Sec. 3.4. The intensity λ is the parameter of the stochastic process. As a complete estimation by a global method is not possible due to the incremental update of the environment, another reasonable estimator can be used. The use of Fuzzy Update for estimation of λ is therefore no violation of the probability theory. Based on this λ, entropy can be computed for the Poisson Point Model.

Further, entropy can be described by the occupancy state $p(c_i)$ directly. The computation of the state value $p(c_i)$ depends on the update method described in Sec. 3.6.5:

$$\begin{aligned} p(c_i)_{Bayes} &= \frac{e^{l_{full}}}{1 + e^{l_{full}}} \\ p(c_i)_{Belief} &= \frac{m_{c_i}(occ) - m_{c_i}(free) + 1}{2} \\ p(c_i)_{Fuzzy} &= \frac{m_{c_i}(occ) - m_{c_i}(free) + 1}{2} \end{aligned} \ . \tag{4.12}$$

These values are used to compute

$$p(\mathcal{B}_u) = p(\mathbf{x}) = p(c_i) \ . \tag{4.13}$$

In this case, the probability is directly defined by the occupancy state $p(c_i)$ of the respective cell. The entropy is zero, when the cells are either known to be occupied (meaning an object was scanned) or free.

The IG is defined as the expected entropy reduction. Technically the entropy merely measures the current information. The IG is the expectation of the entropy, i.e.

$$\mathcal{I}(c_i) = E[H(c_i)] \ . \tag{4.14}$$

The computation of the IG is more challenging than the calculation of the entropy, as the potential outcome of the sensings must be considered. Similar to the assumptions made for the $\mathcal{C}$-exploration problem in

Chap. 3, the quality of the acquired information is regarded. Assuming that the sensor measures the correct occupancy with the probability $p_{correct} = 1 - P(S)$, with $P(S)$ being the probability for a flawed surface (cf. Sec. 3.3.2), the probability of a correct measurement for *occupied*, i.e. surface p_S, is

$$p_S = p_{correct} \cdot p(c_i) + (1 - p_{correct})(1 - p(c_i)) \tag{4.15}$$

The difference between IG and entropy is not large (cf. Sec. 17.4.1 in [164]), particulary if the sensing quality is assumed to be independent of the sensing distance, i.e. $p_{correct}$ is constant. For a correct evaluation, the full inverse model derived in section Sec. 3.3.2 must be applied when guiding the inspection. As the computation is quite complex, the improvement of the exploration process in relation to the computational cost is questionable.

Most approaches apply a more coarse assumption, denoted binary gain [164]. The binary gain simply differentiates between cells that have been updated once, denoted *explored* $\mathcal{I}(c_i) = 0$, and all other cells $\mathcal{I}(c_i) = 1$. This method is applied in frontier-based exploration [52, 53] with mobile robots and works well in practice, as it moves the robot towards the nearest unexplored area. For object inspection, this method is not applicable, because one update of a region does not provide a complete acquisition of an object. Therefore, the higher effort in computing the entropy based on $p(c_i)$ is favored. The inspection process terminates when either $p(c_i) = 1$ or $p(c_i) = 0$ is true for all cells within the RoI.

The IG is computed based on the grey scale values of the $\mathcal{P}$-space, applicable to all update rules. In the following, a different IG computation, applicable specifically to Fuzzy Update, is presented. When Fuzzy Update is applied, the IG can be computed directly from the independent sets $m_{c_i}(free)$ and $m_{c_i}(occ)$ as follows:

$$\mathcal{I}(c_i)_{Fuzzy,c} = 1 - \frac{m_{c_i}(free) + m_{c_i}(occ)}{2} \tag{4.16}$$

This Fuzzy IG is not based on entropy, as this measure is not defined in Fuzzy Theory. Therefore, fusion with $\mathcal{C}$-exploration is not possible, as measures from different theoretical backgrounds cannot be fused. The definition in Eq. (4.16) provides a maximum amount of information on the environment. When both membership functions $m_{c_i}(free)$ and $m_{c_i}(occ)$ are zero, no information on the environment is available, corresponding to the highest IG. The IG is minimal when $m_{c_i}(free) = m_{c_i}(occ) = 1$, denoting the highest contradiction in Fuzzy Update. Then, no further information is available, i.e. the exploration is completed. Another approach is to apply a t-conorm to compute the IG:

$$\mathcal{I}(c_i)_{Fuzzy,t} = t(m_{c_i}(\overline{free}), m_{c_i}(\overline{occ})) \ . \tag{4.17}$$

As these measures contradict with $\mathcal{C}$-space exploration based on entropy, where a Poisson Point Process is used to compute entropy, they are not applied.

4.3.4 Sampling

Similarly to $\mathcal{C}$-space exploration, the success of the exploration in physical space highly depends on sampling a good set of view poses. In physical space exploration, sampling with a $\mathcal{P}$-boundary constraint is reasonable. The success depends on the scanning trajectory in relation to the sensor and its location in relation to the scanning joints.

The view sphere (cf Fig. 4.2) reduces the search space for possible view nodes. However, when applying the beam planning model, numerous ray tracing operations must be considered. In addition, the view sphere imposes a scan orientation perpendicular to the tesselated view sphere surface. In case of turn-table systems, this is no drawback, as the DoFs are limited.

When dealing with non-trivial robots, in which the orientation of the sensors w.r.t. the scanned object is not always accessible due to robot kinematic or $\mathcal{P}$-space constraints, the view sphere must allow for sensor placements inside the sphere.

The planning effort increases notably with the resolution: the simulations in Chap. 6 show the large influence the $\mathcal{P}$-space discretization has on the planning time. Therefore, an approach decoupling modeling and planning is pursued.

A cuboid definition of the RoI is preferred, as it allows the robot to enter the RoI while carving the unknown volume. It allows for decoupling the 3-D modeling from the view planning method. Additionally, an integration with the $\mathcal{C}$-space exploration method is possible.

4.4 Conclusion

This chapter develops an inspection method for a task-directed exploration in physical space. After briefly reviewing existing methods, it is demonstrated that the search space for planning views must be limited.

In inspection, most methods try to optimize views on a priori known objects, denoted model-based inspection. This may be sensible when a CAD-model of an object is available. Then, slight deviations of the real object can be detected. Furthermore, this approach delivers optimal views and an adjusted model. Quality control based on sensory data justifies this approach.

However, in exploration of physical space, this a priori knowledge is not available. Therefore, a model-based inspection is not justifiable, demanding for a model-free method. In order to limit the search space, RoIs for the inspection task are defined. While the definition itself is not within the scope of this thesis, a method using the structure tensor for identification is proposed.

An occlusion-based approach to modeling surfaces and a voxel-based method for modeling are presented. The latter is selected due to its flexibility and suitability in combination with the $\mathcal{C}$-space exploration. View planning can be performed as in $\mathcal{C}$-space exploration by applying the planning models described in Sec. 3.2.3 combined with variable update methods.

The inspection process is driven by an aspired IG maximization. The complexity of using a full model for IG computation is questionable, therefore the use of entropy as IG is applied.

The set of views required for inspection with limited vision in 3-D environments is large. In order to account for the dimensionality of the problem, which impedes a direct computation of the possible view points, a simple sampling strategy applying a $\mathcal{P}_{free}$-boundary constraint is preferred.

5 Multiple Task Exploration

This chapter combines the exploration methods for work space and physical space elaborated in the previous chapters and places them into a universal architecture. While focussing on the exploration of $\mathcal{C}$-space and objects in $\mathcal{P}$-space, the architecture proposed in this section is extendable to other tasks such as inspection and object recognition. The information gain representation is based on entropy.

First, a short summary of related work is given, addressing the potential of a common exploration architecture. The interrelation of tasks is assessed for a flexible work cell; the different challenges are recapitulated separately, elaborating their specific properties. Further, an architecture integrating these task-specific problems and their peculiarities and requirements is presented.

In order to store task-specific information efficiently, the $\mathcal{P}$-space representation is elaborated. The findings in the previous chapters are applied in order to asses the required $\mathcal{P}$-space implementation, as the world model connects the different exploration strategies. The combined $\mathcal{C}$-space and $\mathcal{P}$-space exploration task is presented. The chapter closes with concluding remarks on the architecture and its usability.

5.1 Introduction

Methods for surface modeling and object recognition are not in the scope of this thesis. However, related work at the DLR's Institute of Robotics and Mechatronics enables realistic experiments. Details on inspection are published by Bodenmueller et al. [19], focusing on incremental surface reconstruction from unorganized range data. Wahl

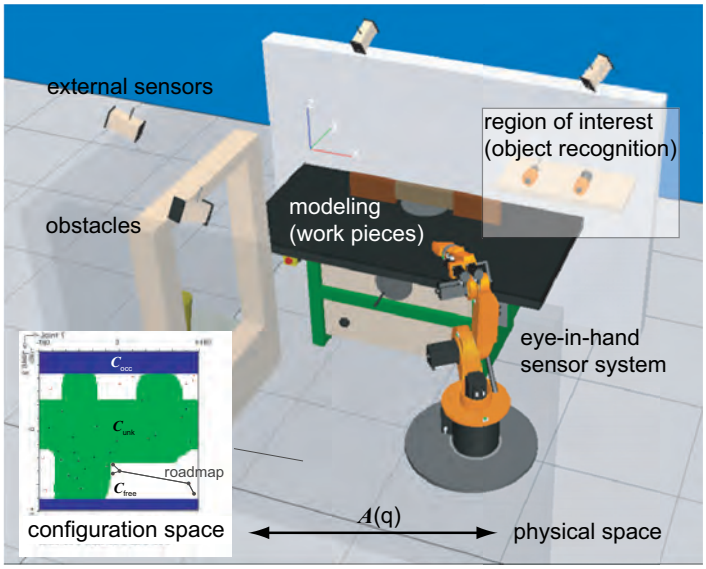


Figure 5.1: Flexible work cell for SME applications requiring exploration, object recognition and modeling: Motion planning is performed in a roadmap. The lower left corner shows a 2-D slice of the $\mathcal{C}$-space. Blue areas represent $\mathcal{C}_{occ}$, green areas describe $\mathcal{C}_{unk}$, and $\mathcal{C}_{free}$ is defined by white areas.

et al. [174] describe a pose-invariant surface-based object recognition, while Ott et al. [121] use point models for recognition, including pose estimation of objects.

In the following, two examples for work on multiple task execution are given: Stilman et al. [148] generate plans for articulated robots; the efficiency is optimized by identifying methods for sampling object surfaces and generating connecting paths between grasps and placements in task space. Uncertainty due to sensor-based operation is not considered. Burns et al. [33] address the major assumption – the world is perfectly known and ideal – for most motion planning methods. They present an operation of manipulators using sampling-based algorithms which consider sensor uncertainty.

A combination of both approaches, i.e. the generation of motion plans for complex robots and sensor uncertainty, is elaborated in the following example of flexible work cell, where multiple task exploration is performed.

5.2 Flexible Work Cell

Today, there is a growing need for flexible automation, i.e. an adaptive planning of motion, changing work pieces, partly unknown work spaces, and human-robot cooperation, especially when robots are used in Small and Medium-sized Enterprises (SMEs). A typical work cell

Figure 5.2: Exploration progress in a work cell using the exploration sensor: Grey blocks define unknown areas, black blocks are known obstacles. Approximately 80% of the configuration space is known.

from such an environment is depicted in Fig. 5.1, shown as a CAD model for better overview. In this cell, the robot is to be set-up automatically, requiring an autonomous exploration of the cell when only partial CAD models of the surroundings are available.

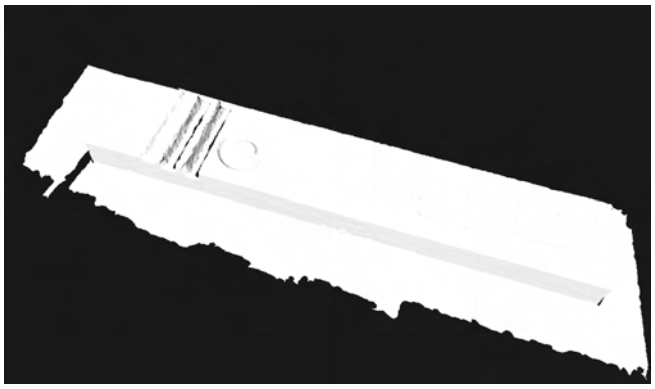
Secondly, work objects are subject to change, e.g. a wooden work piece in a joiner's workshop. In most cases, only limited CAD information is available. Therefore, inspection of these work pieces is necessary in order to acquire CAD models for planning work processes such as spray-painting using the robot. A model of such a work piece is given in Fig. 5.3(a).

Furthermore, tools may be positioned manually in the work cell by workers. In order to perform a successful tool change, recognition and pose estimation of the tools is required.

All of these tasks require a *safe-for-motion* work space model based on a combination of ideal (CAD) and sensory data. In order to clarify the requirements for data containers and hierarchy, the different tasks are briefly recapitulated.

5.2.1 Work Space Exploration

When an articulated robot is installed in a partially known work cell, exploration is performed in order to gain knowledge of its surroundings. A typical work cell in a wood-working application is shown in Fig. 5.1. Initially, the work cell is only partially known, the robot has to gain knowledge of its configuration space and physical space. A sensor is mounted in hand-eye configuration. Sensor results are inferred in an occupancy grid representation; Fig. 5.2 displays the process of such an exploration.



(a) Surface model of a wooden work piece

(b) Triangulated and textured 3-D model of a bust

Figure 5.3: Surface models generated using the surface triangulation method in [19], the wooden work piece model is generated with the robot, the indian bust is acquired manually.

5.2.2 Object Tracking

In case humans enter the work space of a robot in order to interact with the system, e.g to pass work pieces, the sensor can be used for tracking objects [138]. This model-based approach requires a textured 3-D point cloud. The objects can be tracked using only one camera in all 6-DoFs, and grasping with the robot is possible [137].

5.2.3 Object Recognition

In this application the sensor is used for object recognition; it acquires stereo images which are processed using the Multiple Window Multiple Filter (MWMF) stereo algorithm [71] to analyze table-top scenes. Representations of the objects are generated beforehand, then the object is placed on a table in front of the robot. Object recognition and pose estimation is performed.

5.2.4 Object Modeling

Object modeling requires a method for an automatic computation of view points. The desired representation is a surface model suitable for grasp planning. The sensor is applied for 3-D modeling of small objects. The model presented in Fig. 5.3(b) was acquired using the a Laser-Stripe Profiler, described in detail in Chap. 7. The triangulation algorithm [19] generates a 3-D meshed surface of the object during data acquisition. Concurrently, texture information is acquired and later mapped to the surface. Another example is the wooden work piece shown in Fig. 5.3(a).

5.2.5 Summary

In this section, four tasks regularly performed in a flexible work cell in an SME application are described. Work space exploration, object recognition, object tracking, and object modeling must be performed interlacedly, e.g. in order to model objects, the robot must have sufficient knowledge of its maneuverability. The architecture which integrates these tasks is described in the following section.

5.3 Exploration Architecture

Multiple task exploration requires an architecture supporting an interrelation between the system components. Therefore, interfaces between the modules must be defined and data access points must be described. The architecture flowchart presented in Fig. 5.4 depicts a possible approach to such a harmonized integration.

The modules are connected via the world model, i.e. the physical space. It is assumed that in case task-specific information is lacking, this knowledge is obtained through sensor-based exploration. In this thesis, the information gain is defined by entropy reduction, while in general other measures are also possible.

A sensor system is mounted on a robot. Data is acquired and distributed to different modules. The sensor model infers the world model, i.e. the physical space, from the sensings. It incorporates the sensor model, either forward or inverse, depending on the world model (cf. world model implementation in Sec. 5.5).

The exploration process is guided by entropy. Therefore, an $\mathcal{I}$ representation, i.e. IG map, must be represented in $\mathcal{P}$-coordinates. The IG is computed depending on the task, while all views are planned based on the IG map. Based on the $\mathcal{C}$-space of the robot, the NBV is tested for accessability by the motion planner. In case the requested NBV is accessible, motion and sensing are performed. The IG and the following NBV are computed.

The IG can be acquired simultaneously for multiple tasks, e.g. $\mathcal{C}$-IG and RoI-IG, by biasing the NBV computation with a task-specific weight. Object recognition might require a different data representation than motion planning. Therefore, the sensor provides raw range points that are modeled in a task-oriented manner. The world model, being the interface between the tasks, contains an abstract representation of the $\mathcal{P}$-space.

In order to acquire an abstract data representation, the object recognition task accesses the global data base of objects. When an object is recognized, its geometry, e.g a surface-based model, is inferred from the data base and placed in the environment map. If $\mathcal{P}$ is im-

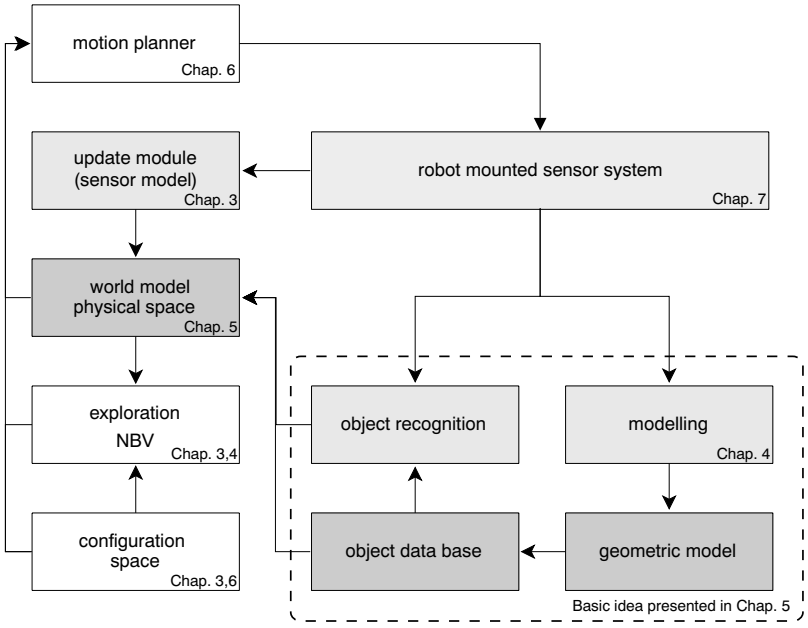


Figure 5.4: Architecture for combining exploration, object recognition and modeling. The architecture is suitable for any sensor-based robotic application and environment. The exploration part is model-free, the data base considers model-based approaches such as object recognition. All tasks can be applied on different data abstraction layers. The part in the dashed box is not part of the thesis, yet initial thoughts on integration are presented in this chapter.

plemented volume-based, the surface-based geometry is voxelized, i.e. represented by bounding volumes. The object recognition itself is not addressed in this thesis. However, methods which are in line with the proposed architecture exist, e.g. Arbel et al. [10].

The object data base can be extended with new objects. In case the object recognition detects a cluster of points belonging to an unidentified object, a surface modeling algorithm can acquire a 3-D model. This model is then included into the object data base. Additionally, RoIs, defined as proposed in the previous chapter, can be used to guide the modeling process.

The architecture combining the tasks, described in the previous section, allows for multiple target exploration. In the next section, the definition of the tasks, their IG computation, and termination criteria are elaborated.

5.4 Task Definition

Most robotic tasks are actually performed in physical space $\mathcal{P}$, whereas motion planning problems are commonly defined in the $\mathcal{C}$-space of a robotic system. Geometrically, $\mathcal{C}$ is mapped to $\mathcal{P}$ by forward kinematics and the geometry of the robot, denoted $\mathcal{A}$. Three exploration modes are defined:

1. A configuration $\mathbf{q}_{goal} \in \mathcal{C}^i$ is given. If $\mathbf{q}_{goal} \in \mathcal{R}$, a path $\mathcal{L}(\mathbf{q}_i, \mathbf{q}_{goal})$ to access the goal is planned. If $\mathbf{q}_{goal} \in \mathcal{C}_{occ}$, the goal is not accessible, whereas $\mathbf{q}_{goal} \in \mathcal{C}_{unk}$ or $\mathbf{q}_{goal} \in \mathcal{C}_{free} \wedge \mathcal{L}(\mathbf{q}_i, \mathbf{q}_{goal}) \cap \mathcal{C}_{unk}$ allow an exploration of the goal. Next, a weight w_{goal} is assigned to guide the exploration towards the goal. The goal is mapped to the physical space in order to allow an efficient calculation of the NBVs.

2. A RoI is described in $\mathcal{P}$. This region is acquired in order to generate sensor data for object recognition and modeling. This exploration is performed in physical space. For computation of views, the entropy measure is based on good results in $\mathcal{C}$-space exploration. Additionally, following the same approach to inspection and exploration enables a use of the same representation and thus presents a major step towards simultaneously solving several tasks in flexible work cells. The representations for higher levels of abstraction, such as object recognition, are maintained separately.

3. Partial and complete $\mathcal{C}$-space exploration must be differentiated:

   The first aims at achieving a complete knowledge of the robot's maneuverability. This task requires no a priori knowledge except for a small region around the initial robot configuration known to be free. The $\mathcal{C}$-space becomes completely known to the robot, when either all configurations are known, or further unknown regions in $\mathcal{C}$ are not accessible due to blocking obstacles.

   The second requires a definition of $\mathcal{C}_{RoI}$. If the path to the RoI is still unknown, a goal exploration to access the region must be conducted. Subsequently, partial exploration utilizing the termination criteria defined for the complete exploration problem is performed.

The measure to guide these three exploration modes is stored in the same representation, the uniformly tesselated IG map denoted $\mathcal{I}$. As in the previous chapters, the IG is represented in physical space. As termination of exploration is a crucial parameter [136], the three tasks and their individual termination criteria are summarized in Tab. 5.1. The views are planned on the same data representation $\mathcal{I} \subseteq \mathcal{P}$ in all tasks. Thus, all of them can be performed simultaneously. Each task

Table 5.1: Termination parameters for the different exploration modes: Tasks are represented in $\mathcal{I} \subseteq \mathcal{P}$, where the NBV u_{NBV} is planned

Task	IG Definition space	Termination Criterion	IG	Weight
Goal	$\mathcal{C}$	$\mathbf{q}_{goal} \in \mathcal{C}_{known} = \mathcal{C}_{occ} \bigcap \mathcal{C}_{free}$	$\mathcal{I}_{goal}$	w_{goal}
RoI	$\mathcal{P}$	$\mathcal{P}_{RoI} \in \mathcal{P}_{known} = \mathcal{P}_{occ} \bigcap \mathcal{P}_{free}$	$\mathcal{I}_{RoI}$	w_{RoI}
Exploration	$\mathcal{C}_{complete}$	$\mathcal{C}_{unk} = \emptyset$	$\mathcal{I}_{exp}$	w_{exp}
	$\mathcal{C}_{partial}$	$\mathcal{C}_{RoI,unk} = \emptyset$	$\mathcal{I}_{exp,RoI}$	$w_{exp,RoI}$

is weighted, thus biasing the planning towards one of the goals by fusion of information in $\mathcal{I}$. The NBV for a number of i exploration tasks is therefore computed by:

$$s_{max} = \underset{s}{argmax} \sum_{i} w_i \mathcal{I}_i;\ i \in \mathbb{Z}\ . \tag{5.1}$$

The exploration is terminated when either all weights are zero or the task-specific IG becomes zero, e.g. when the RoI, $\mathcal{C}$-space, or goal configuration to be explored becomes known to be free or obstacle.

The tasks are defined. The link between all tasks and modules is the world model, i.e. $\mathcal{P}$-space. In the following sections, a suitable data representation and hierarchy of the $\mathcal{P}$-space, enabling multiple task exploration, is described.

5.5 Physical Space Representation

In this section, details on data representation and hierarchy used for implementing the tasks described in the previous sections are discussed. The physical space representation, suitable for multiple task exploration as it contains all task-specific information on an abstract level, is presented. First, model-free data representations on different levels are introduced. Then, bounding volumes are discussed. Finally, block-based data hierarchies suitable for multiple task exploration are elaborated. Concluding the section, the combined exploration and modeling task is described.

5.5.1 Model-Free Data Representations

In this section, the abstraction layers for different data representations are given, not presuming model-knowledge.

The lowest representation level is a point cloud. As stated in Chap. 2, this representation is well-suited for fast rendering of data sets. Secondly, object recognition is possible [121]. An example scene is shown in Fig. 5.5. The corresponding point cloud for the scene is shown in Fig. 5.6.

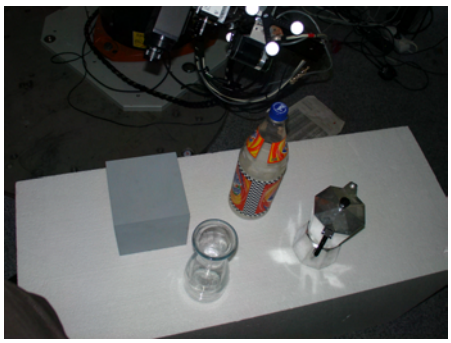


Figure 5.5: Set of work objects with different surface properties: glass, metal, transparent plastic, and solid plastic.

Figure 5.6: Lowest level of abstraction: point cloud representation used for object recognition.

Figure 5.7: Medium level of abstraction: model-free surface representation.

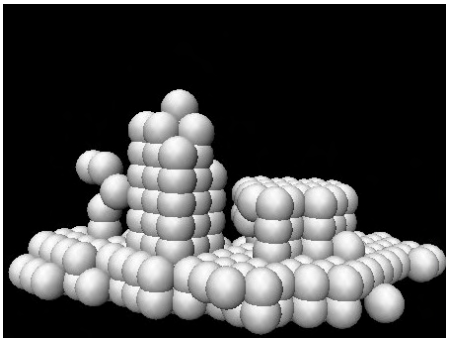


Figure 5.8: High level of abstraction: model-free octree representation as spheres.

For grasping of objects, surface-based representations as displayed in Fig. 5.7 are suitable. The holes and gaps in this particular surface-based representation are caused by non-ideal surfaces such as glass and metal. Thus, planning motions based upon such representations is not safe.

For motion planning based on sensor data, an application of a hierarchy of closed-form primitives is required. In Fig. 5.8, the surface representation is transformed into a sphere tree [25]. This level of data abstraction enables the planning of motions and views.

In this section, the different representations for sensor data are introduced. In the following, the higher abstraction layers for motion and view planning are elaborated.

5.5.2 Representation as Bounding Hulls

The abstraction of the data into bounding hulls enables a representation of different data entities in the same manner. The original outline of the object is lost. This drawback is compensated by advantages in planning views based on primitives instead of surfaces [125]. The most common bounding hulls [31] are

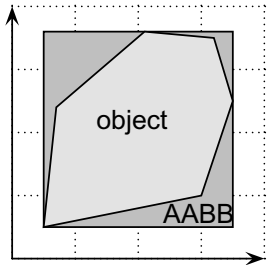


Figure 5.9: Axis-Aligned-Bounding-Box (AABB).

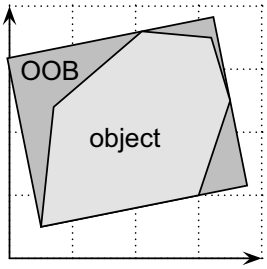


Figure 5.10: Oriented Bounding Box (OBB).

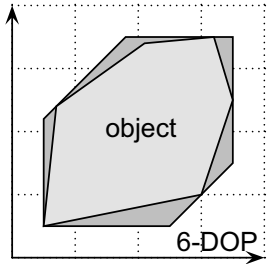


Figure 5.11: 6-Discrete Oriented Polytope (6-DOP).

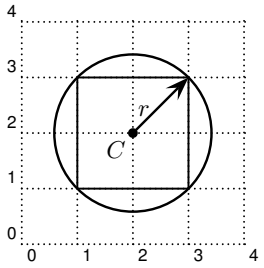


Figure 5.12: Sphere with radius r and enclosed cube.

- Axis-Aligned Bounding Box (AABB);
- Oriented Bounding Box (OBB);
- Discrete Oriented Polytope k-DOP;
- spheres;
- superquadrics.

Fig. 5.9 to Fig. 5.12 show the different bounding hulls. AABBs enclose the object with large overlaps, their computation is trivial. Additionally, they are well-suited for computing views and storing IG, as no orientation must be considered, compensating the less efficient approximation of the real object. In Fig. 5.12, an AABB enclosed by a sphere is given. Spheres have the drawback that they overlap in spatial data structures. Especially when dealing with occupancy grids, overlapping regions require consideration.

For these reasons, the use of AABBs is favored over the other bounding volumes, including spheres.

5.5.3 Data Hierarchy

In the previous section, the use of primitives, e.g. bounding boxes, polytopes, and spheres, is selected for representing sensor data and objects in $\mathcal{P}$-space. In order to enhance the representation, the primitives are spatially organized in several hierarchies. Various data hi-

Table 5.2: Map with indices

Index	Value
$index_0$	$value_0$
$index_1$	$value_1$
$\vdots$	$\vdots$
$index_n$	$value_n$

erarchies are possible; in the following, a single-scale spatial subdivision (voxel map), a multi-scale spatial subdivision (octree map), and a special sphere-tree structure are described. Details are elaborated w.r.t. insertion and deletion of elements, which is often performed in incremental exploration. Spatial data structures are extensively discussed in [130], where locational codes, hierarchies, and implementations mainly for 2-D GIS applications are presented. Implementation details concerning data structures are described in [120].

Voxel Map

In a voxel map, the data is stored as discrete volume elements, so-called voxels. The voxels divide the space into a homogeneous grid with resolution r. Thus, a voxel map preserves the spatial ordering, yet inserted objects can merely be recovered up to voxel space resolution.

While arrays apply integers for indexing, maps lack a numerical index. Therefore, maps are often denoted associative arrays, the indexing is done by coordinates and size of elements.

A spatial index, i.e. locational code, is usually used to access the elements. The coordinates of a voxel are denoted x, y, and z. The size of the respective voxel is defined as s_x, s_y, and s_z, i.e. the voxel map resolution r. In voxel maps, the sizes s_i are identical for all blocks. A locational code lc is computed for single-sized cells using the maximal dimensions of (x, y), denoted (N_x, N_y):

$$lc = x + N_x \cdot y + N_x N_y z \ . \tag{5.2}$$

Maps can easily be represented in tables as shown in Tab. 5.2. The first column contains the index, the second column holds the corresponding date. Furthermore, trees can be used for implementing a map; hash tables present the most common implementation. Depending on the implementation, different complexities for access and insertion arise (cf. Tab. 5.3).

Table 5.3: Complexity of operations in data hierarchies, n denotes the number of elements

Operation	Hash Table	Voxel Map	Octree
Access	$O(1)$ to $O(n)$	$O(\log n)$	$O(\log n)$
Insertion/Deletion	$O(\log n)$	$O(\log n)$	$O(\log n)$

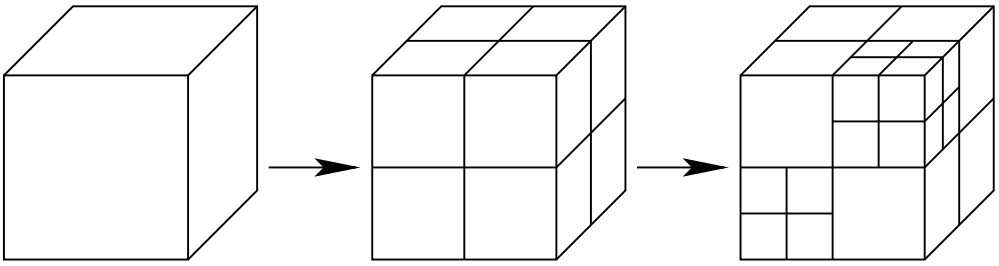


Figure 5.13: Subdivision in octree construction: Blocks are subdivided until either a predefined minimum resolution, i.e. level of detail, is reached or the value of the blocks is identical.

Octree

An octree is a tree structure partitioning the space into cubes or spheres, similar to a voxel map, yet with a voxel hierarchy instead of a single resolution.

The root represents the entire space, and is subdivided into eight smaller cubes. The cubes are recursively subdivided until the new cubes no longer contain additional information (i.e. the entire cube represents the same state), or a minimal tree level (i.e. the minimal resolution) is reached.

An octree subdivision process is depicted in Fig. 5.13. To the left, the root is shown, the subdivision is then applied in two levels. Octrees have the same complexity as the regular trees shown in Tab. 5.3.

Sphere-Tree

The sphere-tree[1] uses spheres rather than cubes as bounding hulls in the leaves. The sphere-tree [24] can be constructed in various ways, approximating the structure of the $\mathcal{P}$-space more or less accurately. The construction of a sphere-tree, which is is analogous to the octree construction, is presented in Fig. 5.14. The voxels are bounded by spheres as depicted in Fig. 5.12. As opposed to a regular octree, collision detection can be performed very efficiently based on sphere-trees.

[1] http://isg.cs.tcd.ie/spheretree/ Link: 2007-02-19

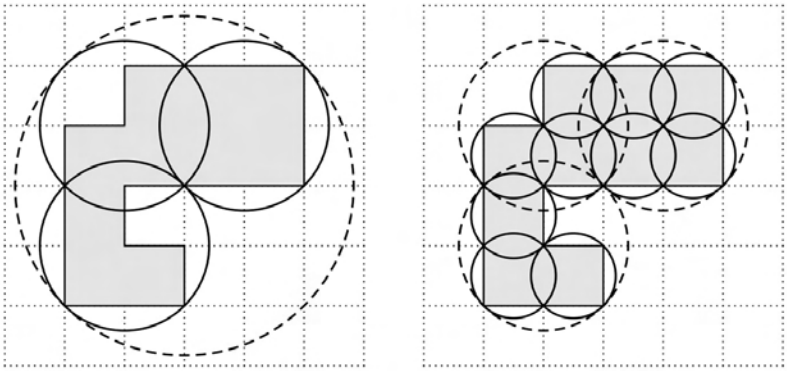


Figure 5.14: Subdivision in sphere-tree construction: The sphere-tree is constructed using an octree algorithm. The voxel blocks are then enclosed by spheres and inserted in the sphere-tree data hierarchy.

Assessment of the Selected Representation

The sphere-tree constructed with an octree method is the most flexible $\mathcal{P}$-space representation for exploration. Points, surfaces, voxel blocks, and spheres can be integrated. Different levels of the tree can be accessed, allowing an application-specific level of detail. Although the octree does not assure the best spatial approximation of the object, it is fastest in construction. Furthermore, a grid-based representation of the occupancy state is applied in exploration. The cubes are inserted into the sphere-tree structure via bounding spheres, as depicted in Fig. 5.12. Incremental construction is possible, deletion and insertion of elements is fast. The data structure is suited for implementation on real robot systems, as a real-time computation of collisions is possible.

The basics for performing multiple task exploration, i.e. architecture, task, definition, and data structure are assessed. In the following, a combined exploration task based on the elaborated prerequisites is presented.

5.6 Integration of Exploration Tasks

The general process model for a simultaneous exploration of work space and physical space, i.e. $\mathcal{C}$-space and $\mathcal{P}$-space, is illustrated in Fig. 5.15. The information about RoIs is given in advance, e.g. a spherical RoI with $\mathbf{c}_{RoI}$ and r_{RoI} is roughly known as depicted in Fig. 5.16. Measures to detect specific features of an interesting area must be defined, e.g. high-edge density or a pre-defined feature ratio. The RoI is conditioned e.g. by passive stereo vision based on edge detection or by preliminary information about the physical work space. Based on this information, an exploration is performed.

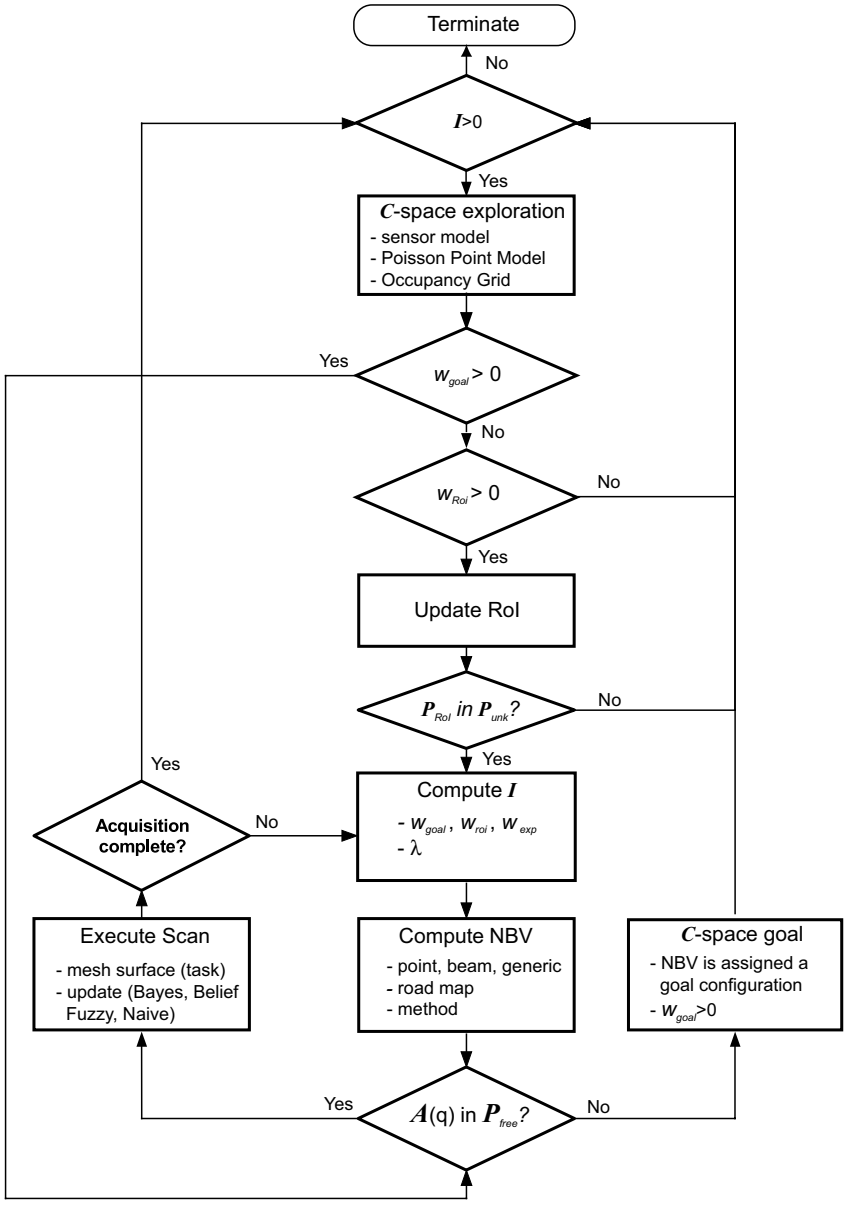


Figure 5.15: Flow chart for the combination of $\mathcal{C}$-space exploration and object modeling tasks in $\mathcal{P}$-space.

Two strategies for the exploration of objects in $\mathcal{P}$-space are discussed in Chap. 4:

1. view point computation based on occlusion and projection of occlusion to spherical RoIs, i.e. a positional space;
2. view point computation based on entropy in cuboid-shaped RoIs considering occlusion.

The view point in the physical space is defined by its location and orientation $\mathbf{x}_{NBV} \in \mathbb{R}^6$, the respective robot configuration in $\mathcal{C}$ is denoted $\mathbf{q}_{NBV}$. In this approach, three cases need consideration:

1. In case $\mathbf{q}_{NBV} \in \mathcal{C}_{free}$ and accessible from the current configuration $\mathbf{q}_{curr}$, the motion planner creates a path connecting $\mathbf{q}_{curr}$ and $\mathbf{q}_{NBV}$ and the scan is performed.

2. If $\mathbf{q}_{NBV} \in \mathcal{C}_{free}$ and $\mathcal{L}(\mathbf{q}_{curr}, \mathbf{q}_{NBV}) = \emptyset$, the goal configuration is free, but no path to the goal exists, requiring exploration. The exploration terminates as soon as the path is recognized to be free, or blocked by obstacles. The latter prohibits a successful execution of the task.

3. When $\mathbf{q}_{NBV} \in \mathcal{C}_{unk}$, the goal configuration is unknown. Exploration is required in order to access the configuration. In order to guide the exploration, w_{goal} weights the goal exploration. The weight for complete exploration is denoted $w_{explore}$. If the goal configuration becomes known, the task can be executed. $\mathbf{q}_{NBV} \in \mathcal{C}_{occ}$ impedes performing the scan.

The following equation describes the computation of the entropy based on the two tasks, i.e. $\mathcal{C}$-space exploration and goal exploration.

$$H(\mathcal{C}) = w_{goal} H(Q_{goal}) + w_{expl} \sum_{i=1..n \wedge i \neq i_{goal}} H(Q) \tag{5.3}$$

The NBV for the object inspection is derived from the physical space directly. While for $\mathcal{C}$-exploration a Poisson Point Process is assumed, the entropy for inspection is derived either from the geometric properties and probabilistic states of the occupancy grid, or from the variation of the Poisson Point density λ.

The combined exploration task requires a representation of the information gain IG. In order to integrate different exploration goals into a common representation, the IG is derived based on entropy measures. The geometric planning methods - point, beam and generalized as presented in Sec. 3.2.3 - are generally applicable for all exploration targets. The planning methods are performed based on the IG map. This map is closely linked to the physical space representation. The resolution of the IG map must be variable to adapt to different physical space resolutions. The IG is represented in a voxel map in order to ensure a consistent computation of the NBV. The view planning method selects the intersecting IG blocks in order to compute the NBV. A multi-scale representation is not suitable, as the intersection cannot determine the true value, e.g. a large block with a low IG value in contrast to a small block with a high IG value.

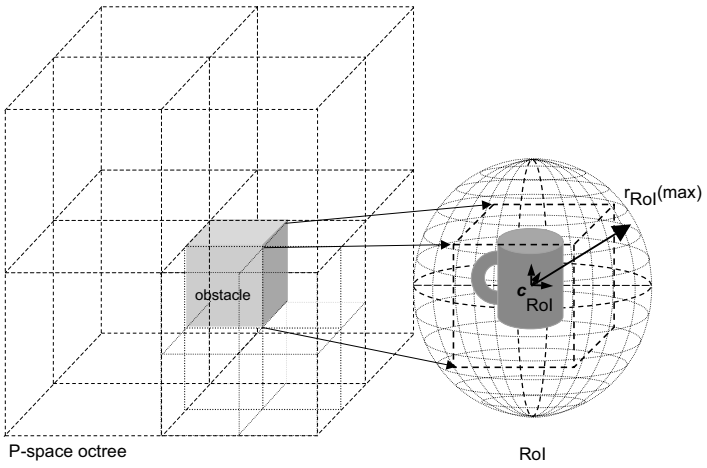


Figure 5.16: Octree implementation with AABB as the bounding volume. The RoI is presented on the right, represented as view sphere ($\mathbf{c}_{RoI}, r_{RoI}$) or block-based ($\mathbf{x}_{RoI}, \mathbf{s}_{RoI}$). The surface data set is kept in a separate data structure, linked to the respective AABB.

5.7 Conclusion

In this chapter, the exploration approaches developed in the previous chapters are combined. The different tasks, i.e. $\mathcal{C}$-exploration, object tracking, object recognition, and object modeling are demonstrated exemplarily for a flexible work cell. The application of robots in SMEs, enabling escape from the automation trap, pushes the development of these methods. Oftentimes, installations of robots in SMEs are planned based on incomplete knowledge on each individual work cell. Most work pieces have non-ideal object properties, and skilled personnel for interacting with, jogging, and teaching of robots is usually not available. Therefore, the methods developed in this thesis apply well for these cells; sensor-based operation is required.

An architecture for combining the exploration tasks is developed. The view planning is driven by a common measure: entropy or - more precisely - information gain. The data structure for representing the environment is presented: A sphere-tree constructed by an octree method is most flexible, the different levels enables an efficient performance of all tasks. In the next chapter, the simulation environment implemented for evaluating sensor properties, data structures, and planning methods are described.

6

Simulation of Exploration Tasks

This chapter presents the simulation environment used for testing and evaluating the exploration and inspection methods developed in Chaps. 3 to 5. In order to perform realistic simulations in 3-D, an environment that incorporates imprecise sensor measurements in 3-D, complex robot kinematics, and 3-D surface models is designed.

This $\mathcal{P}$-space must be flexible in resolution in order to adapt to the versatile application needs, thus a multi-scale representation, which allows for data integration on the most detailed level, is selected. Furthermore, robots with non-trivial geometry and kinematics are simulated. The simulation environment allows an easy interaction with a real robot in order to clearly demonstrate the relevance and transferability of the simulations.

First, initial simulations for considering noise in the planning stage are presented. These simulations are performed in a 2-D planning environment, without integration of uncertain sensing.

Further, the requirements for a 3-D simulation environment are elaborated, focusing on roadmap methods, planning methods, physical space hierarchy, and measurement integration.

As the $\mathcal{P}$-space can be modeled as voxel map or octree, simulations for both implementations are presented in order to elaborate the major differences. The latter data hierarchy is more flexible in application, therefore it is used to assess design rules for the exploration sensor. Additionally, sampling strategy and world model update strategy are evaluated. Concluding the chapter, the results of the simulations are presented and discussed. This assessment leads to design rules for a combined sensing system, and shows the potential and limitations of the transfer of simulations towards realistic experiments.

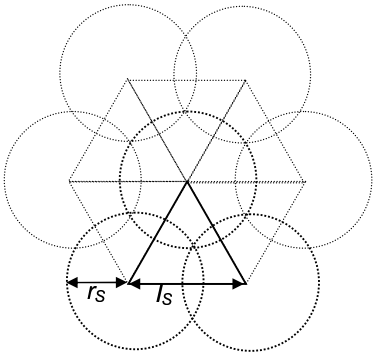


Figure 6.1: Simplex cells: The resolution of this model and thus the diameter of the circles (spheres in 3-D) can be calculated using $l_s = r_s \cdot \sqrt{3}$.

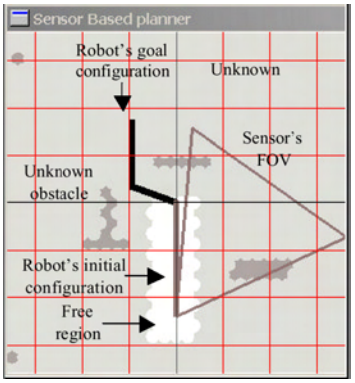


Figure 6.2: Eye-in-hand system: a two-link robot with a wrist-mounted range sensor (with triangular FoV) moving in an unknown environment [159]. The planning tool is owned by SFU-RAMP lab.

6.1 Simulations in 2-D Physical Space

Early experiments in a planar simulation environment, developed at the RAMP lab of Simon Fraser University[1], were performed during a research visit to this laboratory. In order to provide an insight into the basics of $\mathcal{C}$-space entropy as well as sensor-based exploration, the simulation environment and early results for noisy sensor models are presented below. Within this work, experiments in varying sensor parameters, i.e. range, precision, and opening angle, are simulated, applying abstract sensors integrated into the planning environment:

- a short measurement range sensor with high precision (LRS);
- a long measurement range sensor with lower precision (LSP);

6.1.1 2-D Simulation Environment

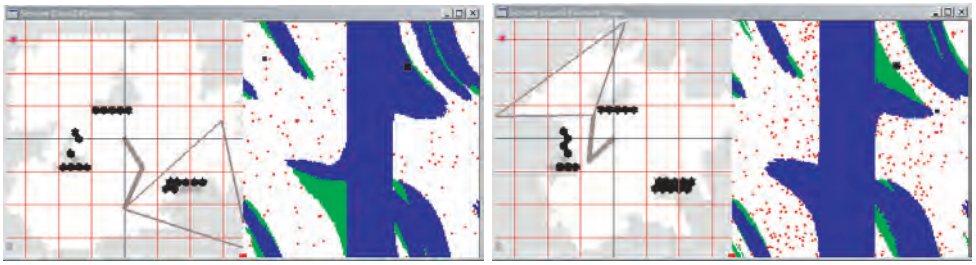
In the environment, it is possible to simulate n-DoF planar robots. The simulation environment as well as its data structures are described in [6]. The $\mathcal{P}$-space is modeled using simplex cells, a data structure similar to sphere-trees in 3-D. The sphere is reduced into a simplex in 2-D. The simplex cell is illustrated in Fig. 6.1. Certain rules must be obeyed while calculating the state of the cells regarding their overlapping nature, e.g. if one cell has the state occ, the neighboring one is $free$, then the overlapping area is defined occ.

The integration of noisy sensings into simplex cells requires consideration of the cell's state in the overlapping region. When using tristate values $p(c_i) = \{occ, free, unk\}$, the state of the overlap is computed

[1] Robotic Algorithms & Motion Planning Laboratory, http://ramp.ensc.sfu.ca/

Table 6.1: Sensor properties used for 2-D simulations

Sensor:	d_{min}	d_{max}	$\Delta\alpha$	$\sigma(d_{min})$	$\sigma(d_{max})$
LRS	50 mm	300 mm	55	0.02 mm	4.73 mm
LSP	80 mm	400 mm	30	0.05 mm	15.00 mm



(a) Noisy MER with LSP (b) Noisy MER with LRS

Figure 6.3: $\mathcal{P}$-space and $\mathcal{C}$-space representation after 18 iterations; the $\mathcal{P}$-space is idealized and modeled in a 2-D grid consisting of simplex cells, shown to the left. On the right, the 2-D $\mathcal{C}$-space is presented; blue (dark) areas denote $\mathcal{C}_{occ}$, green (light) areas depict $\mathcal{C}_{unk}$. The red dots in $\mathcal{C}_{free}$ (white) area show roadmap nodes.

by a deterministic method, e.g. if $p(c_1) = occ$ and $p(c_2) = free$, then $p(c_1) \cap p(c_2) = occ$. For Bayes' Rule this is not possible, because the overlap violates the independence assumption of the cells. Additionally, a single-scale representation causes problems when integrating sensory data. The high resolution required for noisy integration makes computation of free paths and updates of the model intractable, calling for a multi-scale representation.

6.1.2 Simulation with Sensor Noise Models in 2-D

Tab. 6.1 shows the specifications of the two sensor models used in the simulations. The variable $\Delta\alpha$ denotes the opening angle of each sensor. Fig. 6.3 shows snapshots of $\mathcal{P}$-space and $\mathcal{C}$-space after 18 scans. Fig. 6.4 shows the exploration rates of the two sensor models. The noisy exploration rates, i.e. $\mathcal{C}_{known}$, do not vary much from the ideal ones [178], as the noise level is low in relation to the resolution needed for gross motion planning.

In order to reliably validate these results in large-scale environments, a new simulation environment must be developed. The incorporation of noisy measurements, an extension to 3-D work spaces and an integration of real robots are required. In the following section, the key concepts for this environment are described.

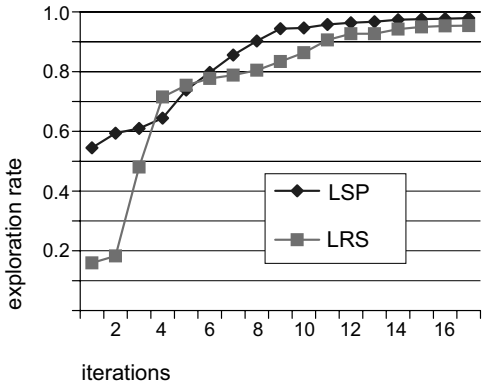


Figure 6.4: Comparison of exploration results: Known $\mathcal{C}$-space percentage vs. number of iterations. The FoV is modeled as presented in Fig. 6.3.

6.2 Simulations in 3-D Physical Space

Within the 2-D environment, the physical space is modeled ideally, however, a representation of sensor uncertainty is required. Additionally, an environment for simulating tasks in physical space is necessary. These and other specifications motivate the development of a simulation suite for a sensor-based exploration of $\mathcal{C}$-space and $\mathcal{P}$-space, which is presented in this section.

In order to directly transfer simulation results to real experiments, the simulation suite implements interfaces to real robot systems.Robots and sensors are exchangeable. The system is represented by its kinematics, its geometry, and its sensor in hand-eye configuration.

Currently, only manipulators with a stationary robot base are considered, an extension to mobile robot systems is anticipated in Chap. 9. The sensor system is defined by the

- sensor type, i.e. laser-range sensor or stereo;
- field of view, i.e. d_{min}, d_{max}, $\Delta\alpha$;
- calibration matrix ${}_{TCP}\mathbf{T}^{sensor}$;
- sensor model.

The sensor models are used for simulating sensor behavior. Additionally, they are required for the integration of measurements, when connected to a real system. The parameters of these models, e.g. Eq. (3.14), are obtained from real sensors by application of Expectation Maximization (EM) algorithms [110].

In the simulation suite, the following modules are required to validate all methods presented in the previous chapters:

1. $\mathcal{P}$-space representation considering uncertainty and information gain;
2. motion planning including navigation and collision detection;
3. kinematics module for forward and inverse computation (cf. [73]);
4. simulation of measurements, independence of measurements and noise simulation;
5. interface to real robot and sensor for simulations and real experiments (cf. Sec. 8.1);
6. visualization of $\mathcal{P}$-space, information gain, robot, and sensor in 3-D;
7. view planning module.

Numerous simulation tools implement subsets of the required modules:

A widely used research tool, originally developed for mobile robots, is Player/Stage[2] and its 3-D extension Gazebo. It contains a robot device interface (Player) and a 3-D multiple robot simulator (Stage/Gazebo). The simulator provides sensor models and robots and is compatible to Player. It does neither implement a view planning module nor a grid-based uncertainty representation in 3-D, which is required in this thesis.

The 3D-Create software's[3] core components are identical to the KUKA.Sim Pro simulation environment. To date, both tools neither allow real sensor data integration nor uncertainty in $\mathcal{P}$-space. They are commonly used for definition and simulation in CAD-modeled work cells.

The Reis Robotics work cell modeling tool ProVis allows for 3-D model integration. However, it lacks the flexibility of implementing view planning methods, as it is mainly designed for Augmented Reality (AR) and process visualization.

As no existing tool complies with the requirements specified above, a novel simulation environment, the SBP-Simulator, is developed. This software tool is used for the simulations and experiments. It is specifically designed according to the requirements for sensor-based exploration. Detailed information on the software is provided in Chap. D of the appendix.

As the required modules are interdependent, the structure depicted in Fig. 6.5 is implemented. The task space is represented in the data hierarchy module and linked to the visualization and collision detection

[2] http://playerstage.sourceforge.net LINK: 2007-08-01
[3] Visual Components Oy, http://www.visualcomponents.com Link: 2007-02-17

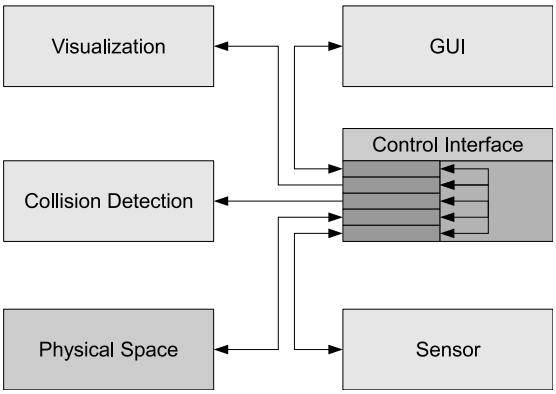


Figure 6.5: Visualization, collision detection, and data hierarchy (physical space octree) are synchronized via an abstract control interface.

via a control interface. This decouples the base data structure from visualization and collision detection, and allows an easy synchronization via observer patterns[4].

6.2.1 Physical Space Implementation

The $\mathcal{P}$-space implementation must be highly flexible, as sensor measurements constantly change the a priori unknown environment. It must represent

- Continuous states of the cells: This is required to represented the uncertainty induced by the sensors. Further, in accordance with Chap. 3 and Chap. 4, the state of a cell c_i is used for computing the intensity of the Poisson Point Process.
- Virtual and real sensings change the physical space. Therefore, efficient methods to access the physical space blocks and to change their corresponding values are required. For single-scale representations, a separate representation, i.e. a high-resolution sensor FoV map, is required for integration of the sensings according to the sensor precision.
- Safe planning of motions: The space is initially partly unknown.
- In simulation, occupied regions are initially partly unknown. Therefore, a simulation state (i.e. the true state of the cell independent of the continuous physical space state induced by the sensings) is required.

In this thesis, two physical space implementations are used: A voxel map and a sphere-tree (cf. Sec. 5.5.3).

[4]The observer pattern (sometimes known as publish/subscribe) is a design pattern used in computer programming to observe the state of an object in a program.

6.2.2 Information Gain Representation

The IG is represented in a voxel map (cf. Sec. 5.5). A single-scale representation is required in order to compute the IG correctly. In the voxel map, the current IG is stored. The resolution is adjustable if the $\mathcal{P}$-space is modeled as a sphere-tree, accessing the different levels of detail.

In the following, the construction of the IG map is exemplarily demonstrated for the task of $\mathcal{C}$-space exploration. The exploration goals in physical space are constructed in analogy, as the IG is represented in $\mathcal{P}$-space coordinates.

Each view planning strategy constructs an IG map as a measure to determine which volumes of an unknown physical space are worth exploring. The IG map is a 3-D histogram over physical space, with each block having an associated value which indicates the approximate likelihood of a configuration intersecting that block. Revealing these areas of unknown space is expected to yield the most useful IG about the robot's works pace. In $\mathcal{C}$-space exploration, this map cannot be calculated accurately in a closed-form solution, as this would require a complete computation of the $\mathcal{C}$-space. Instead, a random sampling of unknown configurations is stored in a list $\mathcal{R}_{unk}$ to build the IG map incrementally. $\mathcal{R}_{unk}$ contains a number of unknown configurations, which are not connected.

Algorithm 1 Computing the IG map for $\mathcal{C}$-space exploration

```
for all grey nodes q_unk in R_unk do
  frac = fraction of A(q_unk) in P_unk;
  for all P-space blocks P that intersect with A(q_unk) do
    compute ig based on strategy;
    B.value += ig; /* B is the corresponding IG block of P, which is
    created if it does not yet exist */
  end for
end for
```

Fig. 6.6 shows the $\mathcal{P}$-space with two blocks of unknown volume, flanking a Kuka robot positioned at the center. It also shows the IG map as well as the randomly sampled grey configurations upon which it is computed. Lighter areas denote regions of higher IG.

The more grey nodes are sampled to create an IG map, the more accurate it becomes. The mere existence of a good NBV configuration - either in the roadmap or in the robot's entire $\mathcal{C}$-space - is not sufficient to guarantee that this configuration will be selected. The probability of successfully selecting the NBV configuration from the roadmap, or computing it directly, increases with the number of grey configurations placed. It is possible that, although many of the roadmap configurations are oriented to scan regions of $\mathcal{P}_{unk}$ with high IG, none of the

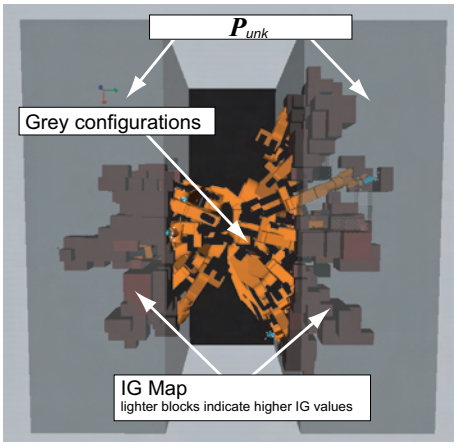


Figure 6.6: IG map and grey configurations overlayed on $\mathcal{P}$-space (top view).

configurations in $\mathcal{L}_g$ intersect with the same regions. Therefore, these view configurations would be deemed to have no value in terms of IG. Even when calculating an NBV explicitly, assuming the extreme case of sampling only one grey node, the selected NBV configuration will scan the volume occupied by this grey node even though it may not result in a relevant IG. Thus, in all view planning strategies, if an NBV configuration is not found, new grey nodes are placed and the IG map is regenerated. This cycle is repeated for a user-determined number of times or until an ideal NBV configuration is found.

6.2.3 Visibility-Based Roadmap

Robots with n-DoFs require efficient representations for navigation in $\mathcal{C}$-space. Complete methods provide elegant solutions for navigation and planning, but are usually limited to either low-dimensional problems or to problems with a special structure [91]. Therefore, sampling-based methods for solving the $\mathcal{C}$-space exploration problem are chosen.

As the environment is initially unknown, a probabilistic approach is taken. The navigation is done in a so-called roadmap, i.e. a 1-D graph $\mathcal{R}$, with vertices representing points in the $\mathbb{R}^n$ $\mathcal{C}$-space and edges representing free paths between these points. The edges can be weighted to assign travel costs. Within this roadmap, the robot can plan and travel the path.

A visibility-based PRM as described in [115] is implemented as the underlying structure for navigation and exploration in $\mathcal{C}$-space. Compared to other PRM construction methods, it may require more time to build, but does produce a relatively sparse roadmap of configurations whose average unshared visibility domain is maximized. The respective algorithm is briefly summarized as follows:

Let $\mathcal{C}_{free}$ denote the collision-free $\mathcal{C}$-space of the robot and $\mathcal{L}$ denote a local method that computes a path $\mathcal{L}(\mathbf{q}, \mathbf{q}')$ between two configurations $\mathbf{q}$ and $\mathbf{q}'$. The visibility domain of a configuration $\mathbf{q}$ for $\mathcal{L}$ is defined by:

$$Vis_{\mathcal{L}}(\mathbf{q}) = \{\mathbf{q}' \in \mathcal{C}_{free} \text{ such that } \mathcal{L}(\mathbf{q}, \mathbf{q}') \subset \mathcal{C}_{free}\}$$

Algorithm

The algorithm used to construct a visibility-based roadmap iteratively processes two sets of nodes: $Guard$ and $Connection$. The nodes of $Guard$ that belong to the same connected component are grouped in subsets G_i. The original algorithm is displayed in Alg. 2.

Algorithm 2 Visibility-based roadmap [115] construction

$Guard \leftarrow \emptyset$; $Connection \leftarrow \emptyset$; $ntry \leftarrow 0$
while $ntry < ntry_{max}$ **do**
  Select a random free configuration $\mathbf{q}$
  $g_{vis} \leftarrow \emptyset$; $G_{vis} \leftarrow \emptyset$
  for all components G_i of $Guard$ **do**
    $found \leftarrow FALSE$
    for all nodes g of G_i **do**
      if $\mathbf{q}$ belongs to $Vis(g)$ **then**
        $found \leftarrow TRUE$
        if $g_{vis} = \emptyset$ **then**
          $g_{vis} \leftarrow g$; $G_{vis} \leftarrow G_i$
        else
          /*q is a connection node*/
          Add $\mathbf{q}$ to $Connection$
          Create edges ($\mathbf{q}$,g) and ($\mathbf{q}$,g_{vis})
          Merge components G_{vis} and G_i
        end if
      end if
      if $found = true$ **then**
        break
      end if
    end for
    if $g_{vis} = \emptyset$ **then**
      /* q is a guard node */
      Add $\mathbf{q}$ to $Guard$
      $ntry \leftarrow 0$
    else
      $ntry \leftarrow ntry + 1$
    end if
  end for
end while

A randomly selected collision-free configuration $\mathbf{q}$ can be handled in

one of three possible ways:

- insertion into the roadmap as a new guard node: $\mathbf{q}$ is not within the visibility range of any existing guard node;
- insertion into the roadmap as a new connection node: $\mathbf{q}$ is visible to two guard nodes from disconnected guard groups G_i and G_j (where $i \neq j$);
- rejection: $\mathbf{q}$ is visible to only one guard group.

The resulting roadmap has fewer configurations and fewer edges than a standard PRM, allowing faster path planning and requiring less memory. The visibility nature of the roadmap is ideal as the optimal configurations and sensor poses for view nodes are typically too specific to be captured in the roadmap via random sampling, particularly in high-dimensional configuration spaces. For n-DoF simulations, a roadmap structure that maximizes the probability of any collision-free configuration (i.e. an explicitly calculated NBV configuration) being connectable or addable to the existing roadmap is applied. This is a key feature of the visibility-based roadmap.

Modification to Increase Sparsity

In order to increase sparsity, an adjustment to the algorithm checks newly discovered guard nodes for connectivity with existing connection nodes. If a connection is possible, this connection is merged into the guard group that the connection node is attached to. In this manner, a minimum number of disjoint guard groups with a minimum number of nodes can be maintained, which requires fewer nodes. Thus, the modification increases sparsity, while reducing the number of possible findings of view points. This is beneficial for efficient motion planning in high-dimensional $\mathcal{C}$-spaces.

In exploration, a larger set of possible view nodes is desired. On the one hand, a separate set of view configurations, from which the NBV is selected and inserted into $\mathcal{R}$, can be sampled. In this thesis, a solid balance between few, optimal nodes is aspired, thus, the sampling criterion is modified, leading towards the selection of few optimal nodes for exploration.

In order to improve the quality of the set of possible view nodes, a biased sampling is preferred, the roadmap's sparsity is satisfactory without the modification. Two methods in that optimize roadmap node selection are presented in the next section.

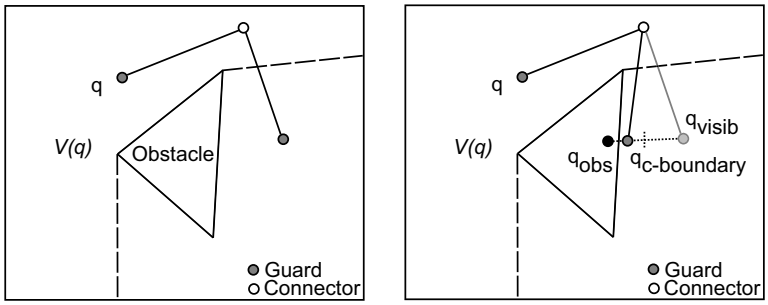


Figure 6.7: Left: Construction of a visibility-based roadmap. Right: Visibility-based roadmap with $\mathcal{C}$-space boundary constraint.

Biased Sampling of Configurations

In PRMs, the sampling method for selecting nodes is crucial for success and coverage of the roadmap view planning method. Hsu et al. [76] state that any kind of bias improving the performance is beneficial, often even required. In exploration, the key interest is to maximally reduce uncertainty with a minimal set of sensings, selected from a large $\mathcal{C}$-space sample, in order to approximate an exact computation of $\mathcal{C}$. Pure randomized sampling in conjunction with a visibility-based roadmap results in a good coverage of $\mathcal{C}_{free}$ with a minimal amount of nodes, enabling optimal path planning, i.e. navigation in the roadmap. Exploration initially requires a high amount of sampled nodes that the optimal view nodes are selected from. This might seem contradictory, yet keeping two sets of nodes, one for view planning and another one for navigation, seems intractable for high-dimensional $\mathcal{C}$-spaces. Therefore, a biased sampling approach satisfying both requirements is applied. Two possible bias are presented in the following.

One bias to the sampling is presented in Fig. 6.7. The distance of the new guard node to all existing occupied and unknown nodes is computed in $\mathcal{C}$-space. The one closest to $\mathbf{q}_{guard}$ is chosen, and the guard node is moved closer under the constraint that the new node is still in the visibility domain of the connection node, i.e. the path is not occluded by obstacles. This so-called known $\mathcal{C}$-space boundary constraint is optimal when exploring with mobile robots, because the $\mathcal{C}$-boundary is closely coupled to the $\mathcal{P}$-boundary due to the trivial geometry of these robots. In this case, the kinematics are trivial, thus nodes located close to unknown nodes indicate good views for performing scans. When exploring with articulated robots, however, this is usually not the case. In Fig. 6.8, the $\mathcal{C}$-space boundary constraint is visualized. The configuration $\mathbf{q} = (q_1, q_2)$ is closest to $\mathbf{q}' = (q_1', q_2')$, the distance is Δq. However, $\mathbf{q}'$ does not provide a configuration which is suitable for exploration using sensors of limited measurement range, i.e. $\mathcal{V} \cap \mathcal{P}_{unk} = \emptyset$. This example demonstrates that the bias is optimal for motion planning, yet view planning requires a different method:

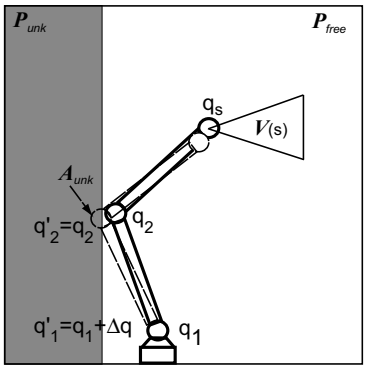


Figure 6.8: Sampling nodes with $\mathcal{C}$-space boundary constraint.

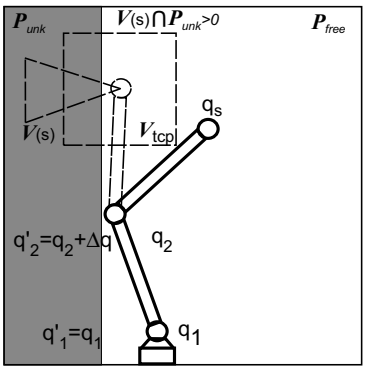


Figure 6.9: Sampling nodes with $\mathcal{P}$-space boundary constraint.

In Fig. 6.9, an alternative bias for sampling under sensor range constraints and non-trivial geometry and kinematics of the robot is presented. In this method, denoted $\mathcal{P}$-boundary constraint, the visibility region $\mathcal{V}_{TCP}$ of the sensor, mounted on the Tool Center Point (TCP), must intersect with $\mathcal{P}_{unk}$. In practice, the bounding box enclosing $\mathcal{V}$ is computed and checked for non-free parts. If $\mathcal{V} \cap \mathcal{P}_{\neg free} = \emptyset$, this node is sampled and passed on to the visibility roadmap construction method. Another possibility to define the visibility region for more general sensors is to compute an AABB, centered at the TCP, with edge lengths $2 \cdot d_{i,max}$, i.e. twice the maximum sensing range. In multisensory mode, $i \in [1, N]$ sensors are available and the sensor with the longest measurement range is chosen.

6.2.4 Collision Detection

A successful determination of the configuration states and that of collision-free paths between configurations, necessary to construct a roadmap of free nodes, requires an efficient collision detection (CD) method. In this thesis, the SOLID library is used. The library is provided with the textbook [171], and implements a fast collision detection based on the Gilbert-Johnson-Keerthi (GJK) algorithm [57]. The Application Interface (API) is similar to OpenGL Graphics System (OpenGL) commands and easy to implement. SOLID keeps an internal representation of the environment, which increases memory consumption and efforts to synchronize the update. In this implementation, SOLID accesses a level of the data hierarchy, therefore a lower resolution can be used for the determination of collision-free paths, while a higher resolution is used for scan updates (cf. 6.5). This technique loosely couples the physical space with the collision inference; a high resolution of the $\mathcal{P}$-space, required for measurement integration, and a lower resolution for the CD, resulting in efficient computation of collision-free paths, can be combined.

The applied surface models for the robot and the sensors are presented in the appendix (cf. Fig. B.1 on page 193).

6.2.5 Path Planning

The path planning is performed within the roadmap and is therefore reduced to path finding. The Djikstra Shortest Path Algorithm [45] is used to determine the traversal path between two nodes of the roadmap. Dijkstra's algorithm keeps the cost $d[n]$ of the current shortest path found between the start node s and each graph node n. The node costs for $d[s] = 0$, all other costs $d[n] = \infty$, as long as no knowledge of a path is available.

The method is commonly denoted as a relaxation problem. The estimated length of the shortest path is described by the length of a tension spring. The shortest path is initially overestimated $d[n] = \infty$, meaning the spring is stretched out maximally. The estimated cost is lowered (shorter paths are found) when relaxing the spring. If a shortest path exists, this corresponds to the spring being completely relaxed.

Individual weights can be assigned to each robot joint in order to vary the cost of traversing an edge of the roadmap, depending on the total displacement of each joint. The cost of traversing an edge between configuration q_i and q_j is defined by

$$\text{EdgeCost}_{ij} = \sum_{n=0}^{DOF} W_n \|\mathbf{q}_j(n) - \mathbf{q}_i(n)\|_2$$

where W_n is the relative weight assigned to joint n.

6.2.6 Robot Kinematics

A C++ kinematic toolkit called C-Rob [73] developed at DLR is used to conduct the kinematic calculations of the robot. The generalized internal representation scheme used by C-Rob permits virtually any type of robot to be modeled by means of transmission objects that define joints and links. This major advantage notably increases the flexibility of the simulation environment for any robot.

A forward kinematics module containing a C-Rob robot specification returns the transformations for the link objects in robot base coordinates with respect to the given joint values. These transformations are used to position link objects for collision detection in SOLID and for visualization of the robot. An inverse kinematics module is used to return possible solutions given the world-coordinate transformation of the TCP.

A popular method for describing the kinematics for robot manipulators is Denavit-Hartenberg (DH) parametrization [65], which defines a chain

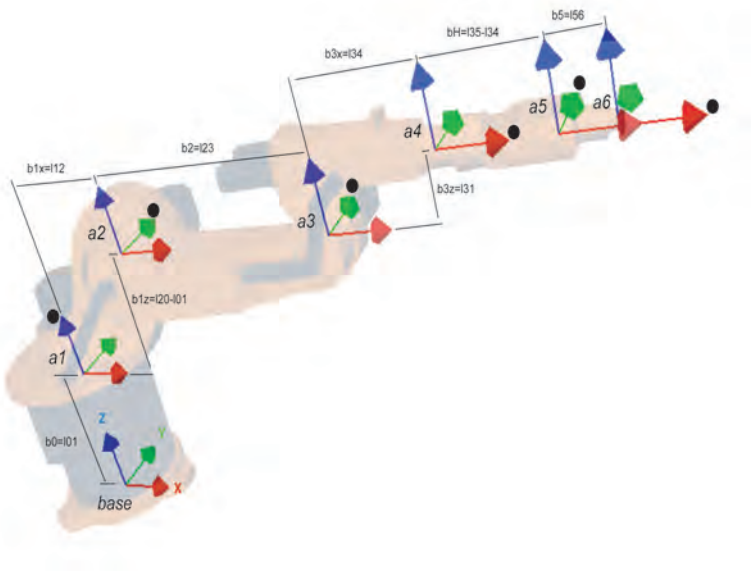


Figure 6.10: Kuka KR6/2 coordinate frames (black dots denote axes of rotation). The parameters of the robot are summarized in Tab. 6.2.

of transformations between the robot links. Although a DH description would be more than sufficient for the robots used in this thesis, given the difficulty of applying this description to more general structures such as kinematic trees and loops (cf. Fig. 7.11), C-Rob utilizes a generic rigid Multi-Body-System (MBS) representation.

Fig. 6.10 shows the C-Rob model of the 6-DoF Kuka (KR6/2)[5] robot in C-Rob home position[6] (i.e. all joints are zero). The coordinate frames for each of the 7 links (denoted by $a0$ to $a6$) have the same orientation in space. C-Rob robot model specifications expect all link frames to be oriented in this manner. Translation vectors (denoted $b0$ to $b5$) define the displacement between the frames. Within the C-Rob Kuka model specification, the frames a are modeled as *CrobRevoluteJoint* objects while the translation vectors b are modeled as *CrobDisplacement* objects. The axes of rotation are defined within the *CrobRevoluteJoints* and are depicted as black dots in Fig. 6.10.

[5]Initial simulations and experiments have been performed with a Kuka KR 6/2 robot. Later, the robot was replaced by a KR16 robot featuring an improved robot controller.

[6]The home position is the initial pose of a robot. In C-Rob, all joint angles are set to zero. For the KR 16 and KR 6/2, the joint values in home are defined as $\mathbf{q}_{home} = [0, -\pi, \pi, 0, 0, 0]$

Table 6.2: Table of joint displacement values

Frame Displacement	$l01$	$l02$	$l12$	$l23$	$l31$	$l34$	$l35$	$l56$
KR6/2 [m]	0.351	0.675	0.3	0.65	0.155	0.32	0.6	0.125
KR16 [m]	0.400	0.675	0.260	0.680	-0.035	0.403	0.670	0.158
Joint Displacement	$b0$	$b1x$	$b1z$	$b2$	$b3x$	$b3z$	bH	$b5$
Length	l01	l12	l02-l01	l23	l34	l31	l35-l34	l56
Joint Offset			$a1$	$a2$	$a3$	$a4$	$a5$	$a6$
KR6/2 [rad]			0	$+\frac{\pi}{2}$	0	0	0	0
KR16 [rad]			0	0	0	0	0	0

6.2.7 Measurement Integration

The sensor measurements must be simulated in a realistic way. In this thesis, beam-based sensor models (cf. Sec. 2.1.2) are used, as these are most similar to the actual physical properties of the sensors. Beam-based sensors usually perform ray tracing in order to compute the effect a measurement has on the environment. If $\mathcal{P}$ is implemented as voxel space of a single resolution, the lower bound of the cell size is defined by memory and computational effort for integrating measurements.

The precision of laser-range scanners ranges from $0.1mm$ to $2mm$, while the physical space has dimensions of $2000mm \times 2000mm \times 2000mm$. In a voxel space of $1mm$ resolution, this results in 2000^3 voxels, even at $25mm$ resolution, 80^3 voxels are required. In order to implement a sufficient resolution for measurement integration, the FoV of the sensor is projected to a high-resolution 2-D map. Ray tracing is performed in this map, then the updated map is projected back into the physical space. This method is prone to aliasing errors as the scale factor between physical space and FoV resolution usually is $s_{\mathcal{P}\rightarrow\mathcal{V}} = \frac{s_\mathcal{P}}{s_\mathcal{V}} = \frac{100mm}{1mm} = 100$.

When using beam-based sensor model implementations in combination with voxel maps or octrees, the cells intersecting the beam must be determined. In voxel maps, all elements are of equal size, while octree elements vary in volume. Therefore, the scan integration technique must be adapted as follows:

Voxel map: Measurements are integrated by determining the cubes intersecting the FoV. In case of a low resolution $s_{\mathcal{P}\rightarrow\mathcal{V}} \geq 100$ and highly accurate sensors, the voxels must be projected in the sensor coordinates in the FoV $\mathcal{V}_{sensor}$, where ray tracing is performed in a map adapted to the sensor precision. After each update, scale adjustments and re-projections are conducted. If the single-scale map features a sufficient resolution, a direct ray tracing in map coordinates is preferred.

Octree: This map contains elements of multiple volumes. Thus, it requires a special procedure for measurement integration. In general, sensors have a higher precision than the map resolution.

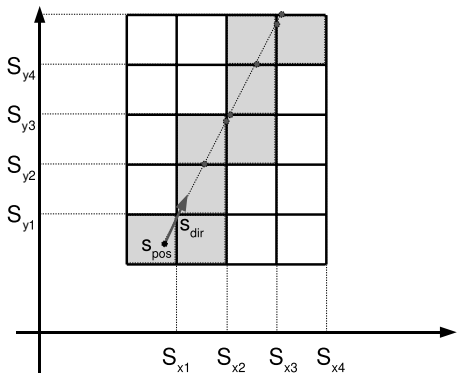


Figure 6.11: Voxel map and computation of cells to be updated by ray tracing in 2-D. The intersections of the cell boundaries $S_{i,x}$ provide integer coordinates of the edges by rounding.

Therefore, measurements are integrated as accurately as possible, i.e. on the lowest level of the octree [130]. Sphere-trees [24] constructed by an octree method provide access to all levels of the tree. Hence, the sensor FoV bounding box accesses the lowest level (the one with highest resolution), providing elements of the same size. This way, the ray tracing can be carried out as described above for a voxel map. After measurement integration, the elements are inserted into the map; merging and subdivision is performed based on the new state.

Subsequently, the ray tracing method verified on single-scale maps can be applied to all $\mathcal{P}$-maps, enhancing the independence of map implementation and scanning method. Methods extracting single-scale maps for the sensor FoV $\mathcal{M}_{FoV} = \{c_i | i = 1, ..n\}$ must be available. Furthermore, axis-aligned voxels are required. In case of projection, the local FoV must be an axis-aligned voxel map. Without loss of generality, the figure depicts the method for 2-D maps, the generalization to 3-D is trivial.

For an efficient implementation of $\mathcal{P}$-space, the coordinate frame F_{FoV} is defined such that the corners of the cells have integer coordinates. Additionally, the edge lengths are normalized to 1 in order to simplify the computation. In this map, cells are accessed by their lower left corner, i.e. the minimal x-, y-, z-coordinate of the cell. The ray tracing method for voxel maps is visualized in Fig. 6.11.

Independent measurements are required for updates based on Dempster-Shafer and Bayes; independence is not necessary for Naïve and Fuzzy Updates. If the independence constraint is violated, regions will become free after having been scanned several times from the same viewing direction. In case of specular surfaces, this produces errors in the map. In [84], the independence of measurements is ensured by keeping a list of view poses and viewing directions for each cell. $\mathcal{P}$-space exploration is performed with a mobile robot equipped

with sonar sensors, the $\mathcal{P}$-space is implemented as a 2-D map. The range of the sensor is divided into three range levels, the angular resolution is divided into 64 segments. Each local cell in $\mathcal{P}$-space contains information on these values.

In this thesis, redundant measurements are recorded in a global map. In order to determine whether a cell in c_i should be updated in the current measurement, a check for redundant a measurement is necessary. Measurements must be differentiated by

- sensor pose $\mathbf{x}_{loc} = [x, y, z] \in \mathcal{P}$;
- sensor view direction $\mathbf{x}_{dir} = [\alpha, \beta] \in [0, 2\pi[, [0, \pi[$;
- measurement distance $d_{dir} \in [0, d_{max}]$.

These values are coded in a global locational code [130]; theses codes are stored in a balanced tree, accessible in $\mathcal{O}(\log\ n)$ with n being the number of elements. Before a measurement is integrated, the map is searched in order to determine duplicate measurements. If the current measurement is identical to an already integrated one, the new measurement is rejected. Sensor pose, view direction, and measurement distance must be discretized to compute the code. The following values are chosen:

$$\begin{aligned} \Delta\mathbf{x}_{loc} &= 20mm \\ \Delta\mathbf{x}_{dir} &= \frac{1}{90}\pi = 2^\circ \\ \Delta d_{dir} &= 5mm \end{aligned} \tag{6.1}$$

For a physical space of 2000mm edge length and a sensor with 500mm maximum sensing range, the number of elements using the discretization in Eq. (6.1) is:

$$n_{ind} = \underbrace{100^3}_{\sim dim(\mathcal{P})^3} \cdot \underbrace{180 \cdot 90}_{const} \cdot \underbrace{100}_{\sim s_{max}} = 1.62 \cdot 10^{12}\ . \tag{6.2}$$

This maximal number is never actually reached in the exploration architecture proposed. The robot performs measurements at discrete optimal NBVs. In general, only few NBVs are required. In a typical exploration process of a work cell with $2000mm$ edge length, the robot performs less than 500 single measurements. If the robot constantly measures during motion, the definition of independence must be softened, fewer measurements are used in $\mathcal{P}$-space update.

Each sensor has unique features. In order to harmonize the interaction of sensor and applications, a generic range interface [20, 154] is developed. Sensors are differentiated by their local coordinate system F_{sensor}, which can be cartesian, cylindrical, spherical, or perspective. The goal is to transmit only a minimal set of information, which is the

transformation from TCP to robot base, i.e. the pose of the robot. Further, the transformation from sensor to robot TCP, i.e. the hand-eye calibration of the sensors, and an equidistant 2-D array of depth values with respective quality is provided with the interface. In order to define *safe-for-motion* areas, the quality measure is applied.

6.3 Implementation of Next-Best-View Methods

In this section, the peculiarities of the implementation of the NBV methods used for the simulations and experiments are described. Apart from the point method presented in Sec. 3.6.1, all methods generate view nodes from the underlying roadmap. Further, the properties of the sensor types used in the simulations are described.

6.3.1 P-Space Boundary-Based View Node Selection

In the Point Strategy (cf. Sec. 3.6.1), all IG blocks at the boundary of $\mathcal{P}_{free}$ are examined to find the one with the highest value. A view configuration is computed via inverse kinematics in order to place this block in the center of the sensor FoV. Given an infinite number of possible viewing directions from which to derive inverse kinematic solutions, only a maximum of 26 viewing directions are checked. These represent all viewing directions that can be described for a 3-dimensional vector of elements $\mathbf{x}_{dir}$ with discretized values -1, 0, or 1 for each of its dimensions. However, once a collision-free and connectable view configuration is found for any given IG block, the remaining view directions are not tested. The basic algorithm is shown in Alg. 3.

Algorithm 3 View node selection based on $\mathcal{P}$-boundary

```
maxValue = 0
for all B ∈ I at the boundary of P_free do
  if I(B) > maxValue then
    maxValue = I(B);
    I_max = B;
  end if
end for
if view configuration q_v can be placed on I_max then
  NBV = q_v;
end if
```

This strategy explicitly computes an NBV configuration, adding a node to the roadmap by force, meaning that it is not in accordance with the visibility-based PRM method for inserting nodes. As connectivity with the roadmap is required, the roadmap constantly grows when using this planning strategy.

6.3.2 Roadmap-Based View Node Selection

In this strategy, the configurations of the existing roadmap are evaluated as potential NBV configurations. This implementation is typically faster, but the probability of failing to find an acceptable view node is high, especially when using a visibility roadmap which is sparse by nature. Therefore, one DoF of the robot is assigned as sensor DoF; a sweep with this joint enlarges the search space. The method is presented in Alg. 4.

Algorithm 4 View node selection based on roadmap

```
maxValue = 0; NBV = NULL;
for all configurations q_v ∈ R do
  if I(q_v) > maxValue then
    maxValue = I(q_v);
    NBV = q_v;
  end if
end for
if NBV = NULL then
  Place new grey nodes
  Resample Roadmap
end if
```

6.3.3 Sensor Types

In this section, an overview on the physical properties of the sensors is given. Three different sensors are regarded; their implementation is described in Chap. 7 of this thesis.

All sensors lose precision with an increasing measurement distance. Additionally, the variance enlarges, which means that the amount of measurement outliers rises as well. In this section, only triangulation-based sensors are considered. Three different abstract sensors are described:

LRS: The first one has a very limited sensing range, but high accuracy. In the following, this type of sensor is denoted LRS, as its characteristics are typical for a triangulation-based laser-range scanner. It applies an active illumination of the scene; a quality value is assigned to each measurement point, describing the intensity of the illumination.

LSP: The second sensor's maximum sensing range is longer, while its accuracy is lower. It is denoted LSP in the following, relating it to be an image-based laser-stripe profiler. It also illuminates the scene actively, however, a quality value per point is not available.

Table 6.3: Sensor types evaluated in the simulation. The sensors are differentiated by FoV dimensions and confidence of the measurement

Sensor	LRS	LSP	SCS
range d	short	mid	large
range quality Q	yes	no	yes
FoV $\mathcal{V}$	2-D	2-D	2.5-D
range confidence $\mathcal{V}_{free}$	yes	no	no

SCS: The third sensor type measures 2.5-D information at a large range for gross information gathering. A stereo camera sensor is a typical realization of that kind. It can be passive or active.

The LRS provides a quality measure with each sensing, which allows a secure determination of *safe-for-motion* regions. Therefore, the LRS assigns the FoV as part of $\mathcal{P}_{free}$ after performing the measurement, if no obstacle is detected in the sensing beam. LSP and SCS must sense an obstacle in order to be able to assign free space in front of it.

The three abstract sensor types are evaluated in simulations in order to deduce design criteria and possible sensor combinations for sensor development. Additionally, a verification of the planning methods is performed. Furthermore, the world model update strategy is evaluated. The sensor preciseness influences the number of updates required for assigning a $\mathcal{P}_{free}$. In order to allow a direct comparison of simulation results with real experiments, the simulation environment integrates interfaces to robots and peripherals, i.e. sensors.

6.3.4 Planning Methods

In order to select the most suitable planning method, a number of approaches must be evaluated, each being applicable for $\mathcal{C}$- as well as $\mathcal{P}$-space exploration.

In the following, the AOD method is applied (Naïve Update applies an intensity $\lambda = 0.5 = const.$, i.e. a homogenous Poisson Point Model).

In order to extend the possibility of finding a view node when selecting view nodes from the roadmap, one DoF, q_{sweep} of the robot is assiged to the sensor, denoted by S in the name of the methods.

The methods are summarized in Tab. 6.4 and detailed in the following.

The Point method (P) computes the $\mathcal{P}$-space coordinates with highest IG on the $\partial\mathcal{P}$, the boundary between known to be free and unknown $\mathcal{P}$-space. Then, attempts to place the FoV midpoint, i.e. $\frac{d_{max}}{2}$ in sensor coordinates, on the respective IG cell are performed.

The Maximum Physical Volume Roadmap Sweep (MPVRS) method selects nodes from the roadmap and tries to maximize the $\mathcal{V}_{unk}$, i.e. the

Table 6.4: Parameters of the view planning methods

Method	Abbreviation	View Node Selection	Sweep DoF
Point	P	$\partial\mathcal{P}$	No
Maximum Physical Volume Sweep	MPVRS	$\mathcal{R}$	Yes
Beam Roadmap Sweep	BRS	$\mathcal{R}$	Yes
Beam Noise Roadmap Sweep	BNRS	$\mathcal{R}$	Yes
Point Beam Sweep	PBS	$\partial\mathcal{P} + \mathcal{R}$	No +Yes

FoV part in $\mathcal{P}_{unk}$ per view node, in order to compute the NBV. Occlusion is considered in the implementation, the method is described in Sec. 3.2.2.

The Beam Roadmap Sweep (BRS) method selects view nodes from the roadmap and computes the NBV based on beam planning model. It applies the ideal beam planning method (cf. Sec. 3.6.2) for MER computation, yet the AOD extension considers uncertainty in the planning stage. The use of BRS with Naïve Update is identical to the ideal beam method w.r.t to $\mathcal{C}$-space exploration.

The Beam Noise Roadmap Sweep (BNRS) method uses the noisy beam method described in Sec. 3.6.3.

The Point Beam roadmap Sweep (PBS) method tries to compute the NBV based on the Point method. If this is not possible, the BRS method is used to compute an NBV. Point tries to place the FoV midpoint on $\mathcal{I}_{max}$ locations. The view directions are discretized. While this limitation is computationally very efficient, a view node might not be found due to the view direction constraints. The application of the beam planning model instead of the point planning model for direct placement of a view node might be more beneficial, yet computationally far more complex. Therefore, in PBS, Point is combined with the sampling-based BRS method, which computes the complete beam efficiently.

6.4 Simulation Results

This section presents the simulation results. First, simulations in voxel space, comparing the noisy beam strategy in Sec. 3.6.3 with the ideal beam method, are described.

The sphere-tree is used as $\mathcal{P}$-space implementation for further assessments. The simulations are performed in order to obtain crucial parameters for the design of the sensor system presented in Chap. 7. The definition of *safe-for-motion* areas has highest priority. All planning strategies apply the Adaptive Obstacle Density method (cf. Sec. 3.6.5) considering uncertainty in the planning. The Naïve Update represents the ideal planning case, i.e. a homogeneous Poisson Point Process with intensity $\lambda = 0.5$.

Figure 6.12: Typical joiner's work bench in an SME wood-working scenario. The picture was taken at Schreinerei Som.

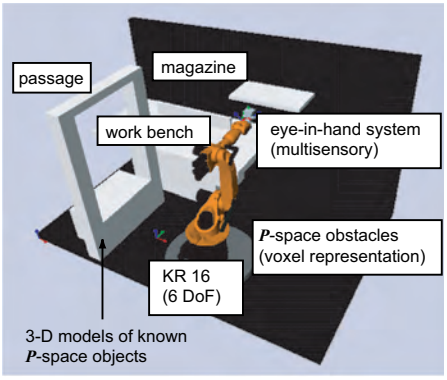


Figure 6.13: Physical space with obstacles simulating a typical SME work cell.

Additionally, different sampling techniques are validated. The influence of the update rules on the exploration performance is assessed. Furthermore, the impact of different scales for $\mathcal{I}$ and $\mathcal{P}$ is evaluated. Simulations with different surface materials are verified with real experiments.

For reasons of comparability, the naïve update is always carried out in parallel, demonstrating the dangers of assuming idealized sensing in sensor-based motion planning. Additionally, the selection of the DoFs assigned to the robot and the sensors is evaluated. Design rules for an optimal *exploratory robot system* are derived.

6.4.1 Comparison of Noisy and Ideal Beam Method

Simulations in $\mathcal{P}$-space represented as a voxel map are presented. The work cell depicted in Fig. 6.13 reflects impressions from a real-world set-up in a joiners workshop. Typical tools like the work bench in Fig. 6.12 are modeled in order to simulate a wood-working environment. The modeled work cell in the simulation environment is depicted in Fig. 6.13.

In general, a priori knowledge can be introduced in two ways:

- regions are set $free, occ, unk$ by defining a box with lower left coordinate and size;
- a surface-based model defined by geometry, location, orientation, and scale, is read in.

If the models are located in $\mathcal{P}_{known}$, they are considered as occupied regions $\mathcal{P}_{occ}$. As the environment is modeled as blocks, the objects are voxelized and introduced into the $\mathcal{P}$-space data structure.

The work cell layout depicted in Fig. 6.13 provides the ground truth, i.e. a CAD-modeled, entirely known cell. In order to perform explo-

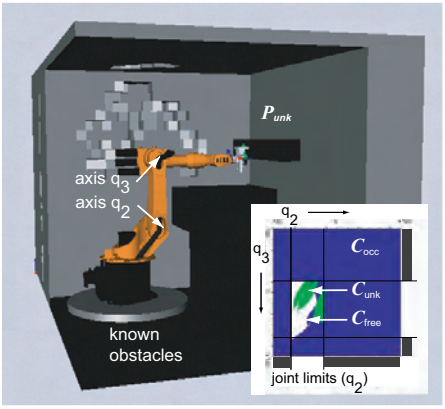


Figure 6.14: Physical space with unknown areas in voxel space representation. Lower right corner shows the $\mathcal{C}$-space: $\mathcal{C}(q_2, q_3) = \mathcal{C}_{unk} + \mathcal{C}_{occ} + \mathcal{C}_{free}$. The joint limits, responsible for the majority of $\mathcal{C}_{occ}$, resemble conditions in a real work cell.

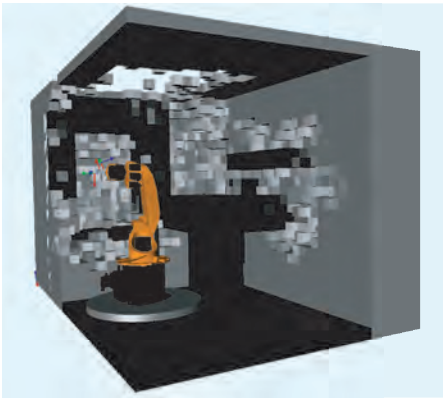


Figure 6.15: Work space (voxel space) after 480 iterations, i.e. attempts to plan a view using the LSP.

ration, parts of the cell are assumed to be unknown. This leads to the representation shown in Fig. 6.14. In the lower right corner, a 2-D slice of the configuration space is depicted. The joint limits of the robot as described in Tab. 6.7 result in large occupied parts in $\mathcal{C}$. This influence of the joint limits on the exploration performance is shown in the next section.

In the following, results of two different exploration methods are compared. Usually, sensing is assumed ideal in standard motion planning methods. In order to account for uncertainties, safety margins are introduced into each motion planning step. In this thesis, uncertainty is already considered in the planning stage; exploration results for the noisy MER method (cf. Sec. 3.6.3) and for planning with the ideal beam method (cf. Sec. 3.6.2) under consideration of safety margins are presented in Fig. 6.16. The $\mathcal{P}$-space resolution is $100mm$, a safety margin of 10% is applied. Thus, every voxel block has an edge length of $120mm$. Fig. 6.16 clearly shows that the noisy MER performs better than the ideal MER with safety margins. This justifies the use of uncertainty in the planning stage.

The sphere-tree is used as $\mathcal{P}$-representation in order to assess the exploration parameters in the next sections.

6.4.2 Assessment of Exploration Parameters

In the following, the parameters relevant for designing an exploration sensor are derived. Furthermore, planning strategies (cf. Tab. 6.4) are compared and recommendations for exploration experiments are given.

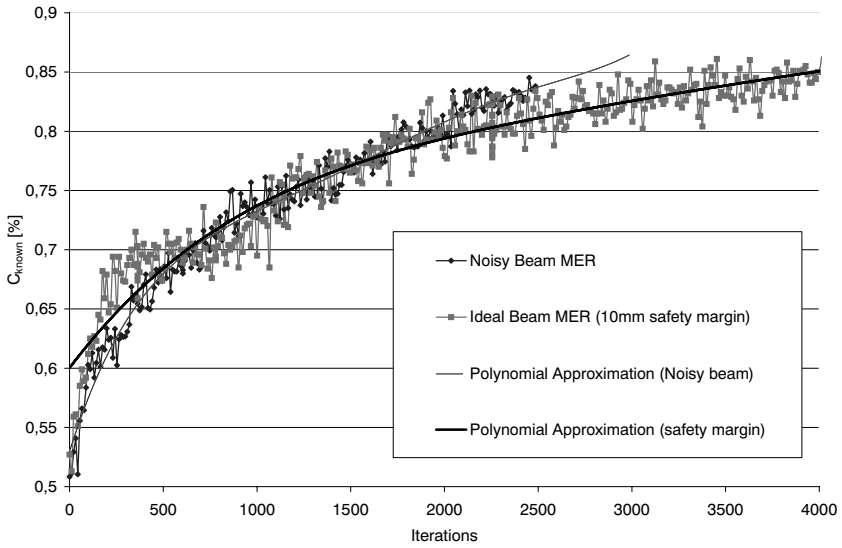


Figure 6.16: Exploration rate for noisy MER and ideal MER with beam planning method

.

Table 6.5: Planning strategies, sensor types, physical space update

View planning strategy	MPVS	BRS	NBRS	P	PBS
Physical space update	Bayes	Fuzzy	Belief	Naïve	
Sensor type	LRS	SCS	LSP		

In the simulations, the sphere-tree structure [23] detailed in Sec. 5.5.3 is used. This hierarchy is denoted PhysicalSphereTreeSpace (PSTS) (cf. Fig. D.2 in the appendix). The structure allows for variable access to $\mathcal{P}$-levels. The collision detection level is denoted $l_{CD} \in [0..6]$, the planning level l_{IG} describes the resolution of $\mathcal{I}$. The resolution is computed by $r_{IG} = 2^{l_{IG}} * r_P$ depending on the $\mathcal{P}$-resolution. Measurements always access $\mathcal{P}$ on the lowest level, i.e. $l_d = 0$.

The simulations are performed with a simulated Kuka KR16 robot, defined by its geometry, kinematics and joint limits. The robot is commanded in $\mathcal{C}$-space, therefore singularities [65] must not be considered in motion planning. The visibility-based roadmap is used for path finding, the parameter n_{rmtry} denotes the maximal number of attempts to place guard nodes in the roadmap, n_{unktry} is the number of tries to place a grey node. In order to determine a grey node, a minimum unknown fraction of $min(\mathcal{A}_{unk}(\mathbf{q})) = \frac{1}{100} \cdot \mathcal{A}(\mathbf{q})$ is defined. The maximum number of grey nodes is denoted n_{unk}, the maximum number of new roadmap nodes per update is denoted n_{rm}. Concluding every iteration i, the values $\mathcal{C}^i_{free}$, $\mathcal{C}^i_{occ}$, and $\mathcal{C}^i_{unk}$ are computed by random sampling. The number of samples is denoted n_{cd}. After planning an NBV, it is

possible that this view node is not accessible. Therefore, a number of s_{max} NBVs are computed per iteration i, accessed and processed in the order of decreasing IG. The four sets of exploration parameters are summarized in Tab. 6.6.

If the roadmap is used for selecting the NBV, however, the chances of finding an NBV is low due to its sparsity (in this thesis, approx. 100 nodes are inserted in $\mathcal{R}$, in [192] the NBV is selected from 10.000 nodes). Therefore, a joint q_{sweep} of the robot is swept in each possible view node. The maximum IG value obtained by moving q_{sweep} for the respective roadmap configuration is computed, resulting in a new NBV. This NBV is inserted into the roadmap.

Furthermore, measurements are performed along a scanning trajectory. 2-D sensors are moved perpendicular to the sensing plane. The trajectory depends on the spatial orientation of the sensor plane in order to ideally acquire 2.5-D information. A 2.5-D sensor does not require sweeping, as it already acquires 2.5-D information. The joint assigned for sensor scans is denoted q_{scan}. In the following simulations, the parameters $q_{scan} = 5$ and $q_{sweep} = 4$ are used for the 6-Dof Kuka KR16 robot.

The scene depicted in Fig. 6.17 and Fig. 6.18 is used to assess the design criteria for a multisensory exploration sensor. The work cell size is shown in Fig. 6.17. The influence of the following parameters on the exploration process must be evaluated and are individually described in the following paragraphs of this section:

- $\mathcal{P}$-space resolution;
- robot joint limits;
- sampling technique and roadmap;
- $\mathcal{P}$-space update rule;
- sensor type;
- view planning method;
- scanning trajectory.

The parameter assessment is used primarily for determining design rules for the sensor system. Further, it supports the preparation of the real-life experiments. Additionally, the simulations enable a verification of the methods developed in this thesis. The figures depicting the $\mathcal{P}$-space, $\mathcal{I}$, and the complete scenes are presented in Chap. B of the appendix.

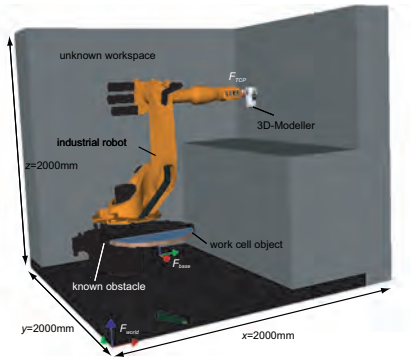


Figure 6.17: Work cell in octree representation.

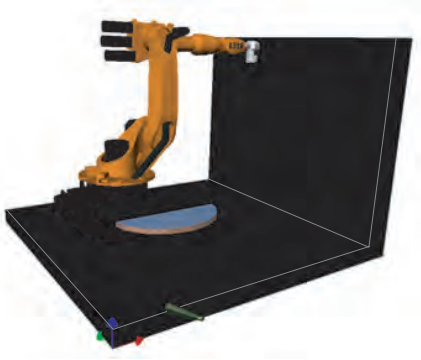


Figure 6.18: Robot physical space with occupied cells.

Table 6.6: Simulation parameter sets for roadmap and physical space resolution

Parameter set	lowres [LR]	midres [MR]	highres [HR]	modeling [MO]
p_{res} [mm]	100	50	25	12
n_{rmtry}	100	100	100	100
n_{unktry}	200	200	200	200
n_{rm}	200	200	200	200
n_{unk}	100	100	100	100
l_{ig}	0	1	1	2
l_{cd}	0	0	1	2
n_{cd}	5000	5000	5000	5000
s_{max}	10	10	10	10

Influence of Physical Space Resolution

In order to evaluate the influence of the $\mathcal{P}$-space resolution on the exploration process, the parameter sets shown in Tab. 6.6 are selected. In this section, the sets [LR] and [MR] are compared. The parameter sets [HR] and [MO] are used in the experiments in Chap. 8.

The $\mathcal{P}$-space Update Type is Naïve, leading to the exploration results depicted in Fig. 6.19. The planning strategy is PBS and the joint limits are set in accordance with the robot's specification, denoted [S] in Tab. 6.7. This eliminates the influence of the joint limits, which is discussed in the following section. The LRS sensor is used for planning views, measurements are performed with LRS, LSP, and SCS in parallel.

In Fig. 6.19(d), it is visible that the influence of the resolution regarding the exploration performance can be neglected. The $\mathcal{I}$-space resolution is identical in both cases, as the IG resolution is defines the planning complexity. In real experiments, the resolution might be more crucial in order to determine the true location of an obstacle in $\mathcal{P}$-space. The $\mathcal{P}$-space resolution affects the memory consumption as well as the measurement integration effort depicted in Fig. 6.19(a), when comparing the total number of updated cells per scan. The total exploration time also increases with the resolution.

Influence of the Robot Joint Limits

Table 6.7: Joint limits for simulations with a Kuka KR16. Three sets are used: [S] Kuka specification, [L] Laboratory specification, and [E] Experimental specification.

Joint	Specification [S]		Laboratory [L]		Experiments [E]	
	q_{min}	q_{max}	q_{min}	q_{max}	q_{min}	q_{max}
q_1	-185°	185°	-110°	55°	-110°	55°
q_2	-155°	35°	-130°	-35°	-130°	-35°
q_3	-130°	154°	35°	135°	35°	135°
q_4	-350°	350°	-350°	350°	-190°	190°
q_5	-130°	130°	-120°	120°	-120°	120°
q_6	-350°	350°	-350°	350°	-190°	190°

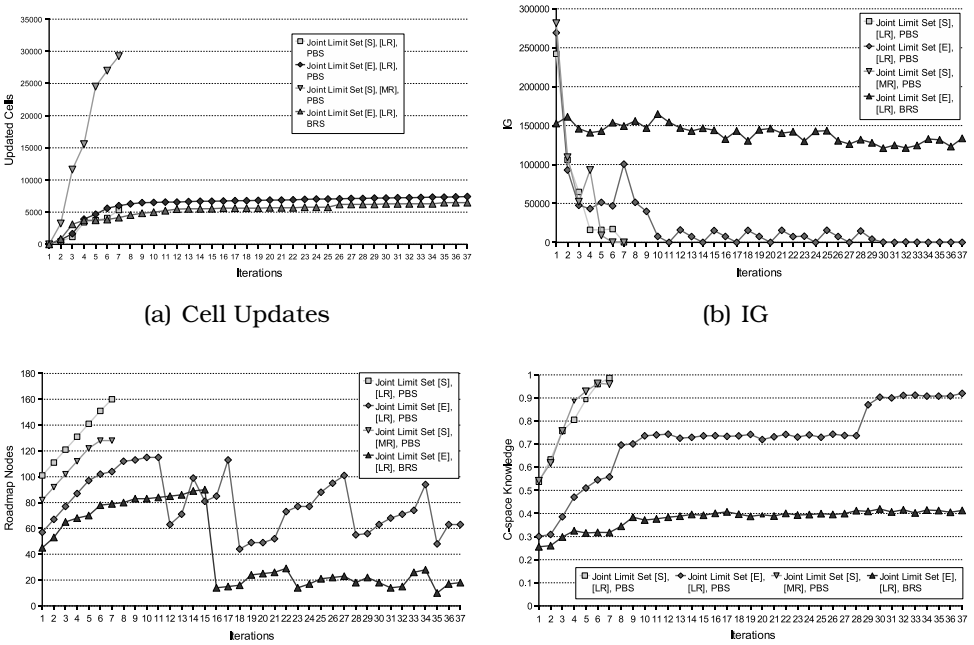
The joint limits used in the simulations and experiments are presented in Tab. 6.7. The first set is derived from the robot's specification. This set [S] is used in simulations in order to determine the influence of the joint limits on the exploration performance. This assessment also provides the lower border for the joint limits, under which exploration is still possible.

The set [L] corresponds to the limitations required for the real test-bed (cf. Chap. 8). The joints q_1 to q_3 must be limited in order to adjust the robot to its limited work space in the installation. These restrictions are used during the simulations for evaluating the impact of joint limits on the exploration process. Joint q_5 is limited in order to avoid clamping of the cables when the sensor is attached to a real system (cables are not modeled in the simulations, i.e. they are neither considered in the kinematics nor in the geometry).

The third set, denoted [E], is defined in order to assign safe conditions for sensor and cabling in the real cell. In addition to q_5, the joints q_4 and q_6 are limited. This is necessary as the sensor's restricted cable length prohibits full turns of these last axes. The accessability remains the same, the only drawback is a reduction of the number of possible roadmap edges.

The influence of the joint limits on the exploration performance is presented in Fig. 6.19(d). The exploration results with PBS for set [S] and set [E] differ notably: While for [S], the task to obtain complete $\mathcal{C}$-knowledge is fulfilled after only 7 iterations, the constrained work space [E] can only be explored in 37 iterations. The robot's accessible $\mathcal{C}$-space is restricted by the joint limits. This results in the difference of the initially known work space for varying joint limit sets, e.g. $\mathcal{C}^0_{known} \approx 0.3$ for [E] is smaller than for [S], where $\mathcal{C}^0_{known} \approx 0.55$ for the same $\mathcal{P}$-space. Some parts of $\mathcal{P}$ might be occupied by the robot, but they cannot be measured by any configuration. Additionally, roadmap edges are straight lines in $\mathcal{C}$, detaining the exploration from accessing a good view node in the roadmap.

The roadmap size is not monotone increasing for PBS and BRS under



(a) Cell Updates (b) IG

(c) Roadmap size (d) $\mathcal{C}_{known}$

Figure 6.19: Assessment of the influence of the joint limits and $\mathcal{P}$-space resolution. The Naïve Update of scene Fig. 6.18; joint limit set [S] and [E] (cf. Tab. 6.7); parameter set [LR] and [MR] (cf. Tab. 6.6).

joint constraints [E] (see Fig. 6.19(c)). This is due to the fact that the roadmap is reset and newly initialized if no NBV is found after performing a roadmap update and re-sampling grey nodes. The roadmap sampling rate is low compared to the dimensionality of $\mathcal{C}$. A possible undersampling is accounted for by the application of the visibility-based PRM, which implements a good coverage of $\mathcal{C}_{free}$. However, the re-sampling for generating a new set of configurations is beneficial after a large number of measurements are have been integrated in the $\mathcal{P}$-space. This results in a better approximation of the coverage of an incrementally enlarged $\mathcal{C}_{free}$ by the roadmap.

Influence of Sampling Technique and Roadmap

For view planning, sampling with a $\mathcal{P}_{free}$-boundary constraint is the best bias. This is justified by the following remarks.

Nature of measurements: Measurements are performed in $\mathcal{P}$-space, therefore sampling of nodes with $\mathcal{P}$-space constraint provides nodes which have a good vision in $\mathcal{P}_{unk}$ and for tasks defined in $\mathcal{C}$ or $\mathcal{P}$, which increases the number of good view nodes selected.

Sampling of $\mathcal{C}$: In large $\mathcal{C}$-spaces, $\mathcal{C}_{unk}$ is approximated by sampling.

In order to provide a good coverage of $\mathcal{C}_{unk}$, oversampling must be avoided, which is not trivial to achieve.

Distances in $\mathcal{C}$ and $\mathcal{P}$: Short distances between $\mathbf{q}_{sample,free}$ and $\mathbf{q}_{unk}$ provide no information on the visibility of the distance with a sensor in hand-eye configuration. The $\mathcal{P}$-space boundary constraint postulates $\mathcal{V}(\mathbf{q}) \bigcap \mathcal{P}_{unk,occ} \neq \emptyset$, therefore the distance between $\mathcal{A}(\mathbf{q}_{sample,free})$ and $\mathcal{P}_{unk}$ is measurable by a sensor in hand-eye configuration.

The above discussions lead to the application of the $\mathcal{P}_{free}$-boundary constraint for sampling nodes. In combination with the assignment of one DoF of the robot to the sensor, a bounding box centered at the TCP with edge length $2d_{max}$ is applied in all simulations, unless otherwise stated.

Influence of the $\mathcal{P}$-space Update Rule

Four update types are evaluated: Bayes, Belief, Fuzzy, and Naïve. The sensor models for measurement integration are presented in Sec. 3.3, their parameters are shown in Tab. 6.8. The influence of the update on the exploration process is depicted in Fig. 6.20.

Bayes' Update applies the inverse model, assuming independence of the cells. Therefore, not all measurements are used for integration. Belief and Fuzzy Update apply the same forward model. While Belief requires independence of the measurement, assured by the global map of measurements, Fuzzy Update does not require independence, every measurement is used to update $\mu_{free}(x)$ and $\mu_{occ}(x)$ respectively. In the following, the findings from the simulations are elaborated.

Fig. 6.20(b) shows that all planning methods achieve an entirely explored $\mathcal{C}$-space. The views are planned based on the LRS with PBS, the measurements are performed with all sensors.

The Naïve Update performs best for exploration as shown in Fig. 6.20(b). In this case, the Poisson Point density $\lambda = 0.5$ is identical for all blocks. Measurement inaccuracies are not considered, the exploration assumes ideal sensing, which results in a minimal number of updates (each cell is either free, unknown, or obstacle after sensing), depicted in Fig. 6.20(a).

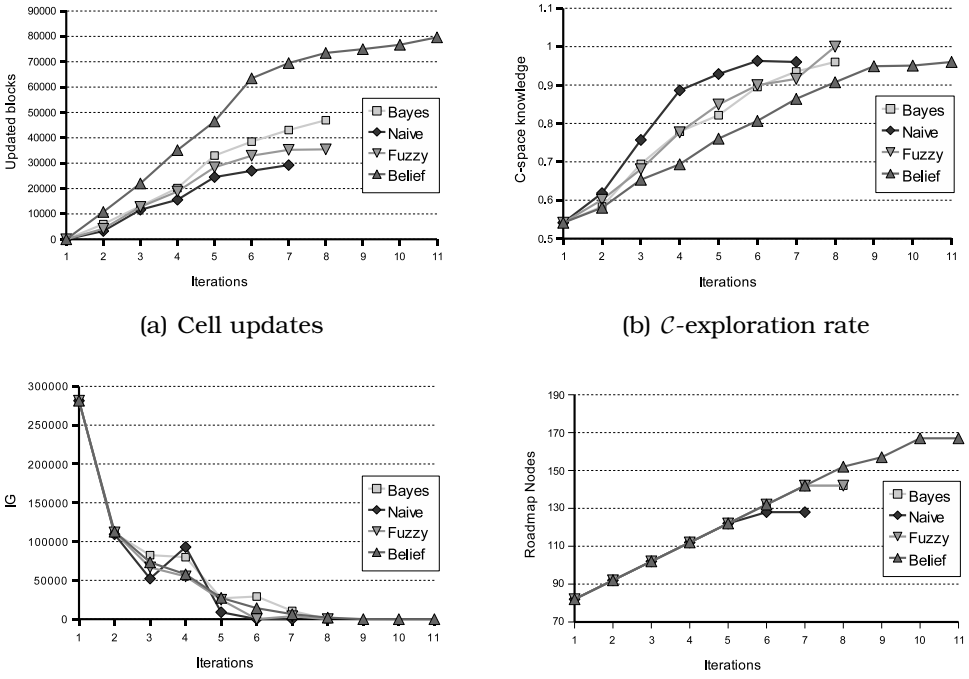
Belief Update performs slowest as its update is most conservative. Cells must be scanned several times (cf. Fig. 6.20(a)). Most cells have been updated in Belief Update, in the current implementation it is most conservative in assigning $\mathcal{P}_{free}$ regions, i.e most cells are updated more than once in order to decide the cell's final state.

Bayes' - and Fuzzy Update perform similarly. In principle, Fuzzy is more conservative in assigning free regions after scans in the current

Table 6.8: Sensor properties for LRS, LSP, and SCS; the variance $\sigma_d(d) = k \cdot d^i$ is assumed to be exponential.

Sensor Type	LRS	LSP	SCS
d_{min} [mm]	60	163	150
d_{max} [mm]	300	463	1000
$\Delta\alpha_x$ [°]	270 (90)	45	45
$\Delta\alpha_y$ [°]	0	0	30
k	$5.52 \cdot 10^{-9}$	$2.0 \cdot 10^{-3}$	$2.0 \cdot 10^{-3}$
i	4.01	1.50	1.50

implementation. However, this drawback is compensated by the required measurement independence for the respective parameter set in Bayes'. This independence requirement virtually reduces the number of measurements performed.



(a) Cell updates

(b) $\mathcal{C}$-exploration rate

(c) IG

(d) Roadmap size

Figure 6.20: Comparison of Bayes', Fuzzy, Belief, and Naïve Update in scene Fig. 6.18, joint limit set [S] in Tab. 6.7, and parameter set [MR] in Tab. 6.6.

Influence of the Sensor Type

Three sensor types are evaluated: LRS, LSP, and SCS. The influence of the sensor type on the exploration is depicted in Fig. 6.21(a) to Fig. 6.21(d). The properties of the abstract sensor types are shown in Tab. 6.3. The LRS is a *safe-for-motion* sensor, as it assigns a quality measure to each measurement. Therefore, regions are assigned free in sensing direction when no obstacle is sensed and the quality value is set.

The complete opposite happens for the LSP. It features a medium range, yet no quality value providing additional information on the sensing. This results in the exploration rates shown in Fig. 6.21(c). Each sensor plans the views based on PBS, the roadmap is sampled with $\mathcal{P}$-boundary constraint and Bayes' Update. The LSP plans NBVs, but no obstacle appears in the FoV. Therefore, no update of the cells is performed, clarified in Fig. 6.21(a).

The SCS suffers from the same constraint, but its longer sensing range reduces the drawback. In a small cell of maximally 2000mm edge length, the measurement range of 1000mm increases the chances for detecting obstacles during the measurements, resulting in an update of the cells (cf. Fig. 6.21(a)) and a reduction of IG (cf. Fig. 6.21(b)).

This is also visible in Fig. 6.21(d): The SCS mainly explores the occupied regions of $\mathcal{P}$-space. In comparison to the SCS, the LSP cannot acquire these obstacles, therefore $\mathcal{C}_{occ}$ for LSP stagnates at a low level. The long sensing range is the main reason for the SCS to explore fastest in the work cell depicted in Fig. 6.18.

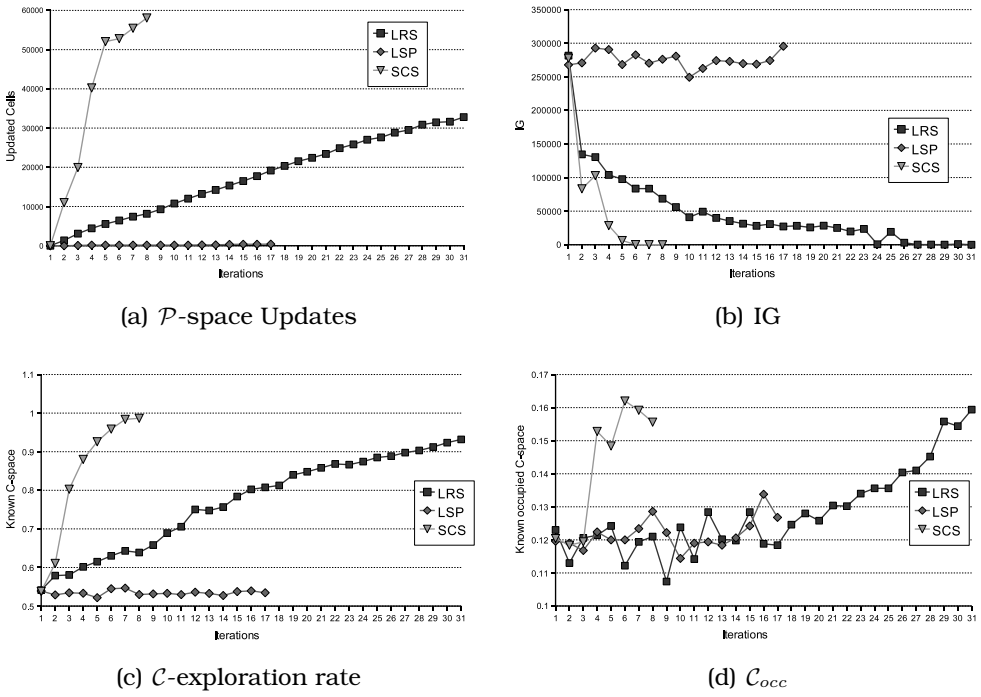
The LRS updates cells in $\mathcal{P}$-space in every iteration. This results in a continuous exploration of the work cell. The short sensing distance dominates the process. However, the LRS is able to achieve a complete knowledge of the cell.

Influence of the View Planning Method

Five planning strategies have been evaluated (cf. Tab. 6.4): MPVS, BNRS, Point, BRS, and PBS.

In general, the Point method does not consider occlusion, thus it is generally less efficient than the BRS method. In this thesis, Point is applied as follows: The cells on the $\mathcal{P}_{free}$-space boundary are considered for placing the FoV midpoint. In doing so, occlusion is limited unless obstacles are present between the current boundary cell and the sensor. However, this rarely occurs when planning with a short-range sensor such as the LRS.

The BRS method uses the roadmap configuration directly and selects nodes from it in order to compute the NBV. This method is beneficial for large $\mathcal{P}$-spaces, where a computation of views directly from the cells



(a) $\mathcal{P}$-space Updates (b) IG

(c) $\mathcal{C}$-exploration rate (d) $\mathcal{C}_{occ}$

Figure 6.21: Comparison of sensor types: LRS, LSP, and SCS with Bayes' Update in scene Fig. 6.18, joint limit set [S] in Tab. 6.7, and parameter set [MR] in Tab. 6.6.

on the $\mathcal{P}$-boundary is not feasible. Additionally, occlusion is modeled and used for planning. However, the roadmap constraint reduces the performance of the method.

BNRS is quite complex in planning, the use of the grey scale values extracted from the map is more beneficial.

MPVS does not improve the IG notably, therefore its application is not recommended. These results are in accordance to [180].

The PBS, i.e. sequential planning with P and BRS, produces the best exploration results, followed by BRS, as shown in Fig. 6.19(d) under joint constraints.

Influence of the Measurement Trajectory

The choice of the measurement trajectory is important. It depends on the pose of the sensor on the robot. In Fig. 6.22, a rotational scan with a slightly eccentrical sensor is presented. The small unmeasured region to the left is caused by the eccentricity of the sensor.

The relation between q_{sweep} and q_{scan} must also be considered; this is depicted in Tab. 6.9 for a KR 16 robot with 6-DoF. In most cases, set A is beneficial, as the sensor is hardly ever mounted centrically regarding the last active joint of the robot. For set A, the possible angular range

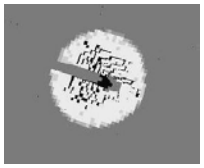


Figure 6.22: $\mathcal{P}$-space changes after a measurement trajectory, rotating $q_{scan} = 6$ with a slightly eccentrical sensor.

Table 6.9: The relation between scan and sweep joint for a Kuka KR16

Set	q_{sweep}	q_{scan}
A	4	5
B	5	6

for joint q_6 is limited to $q_6 \in k \cdot \pi + [-\frac{\pi}{4}, +\frac{\pi}{4}]; k \in \mathbb{Z}$ in order to achieve a 2.5-D FoV based on the measurement trajectory.

6.5 Summary

In this section, the findings from the simulations are summarized; the following results being considered most relevant for a successful sensor design elaborated in Chap. 7:

Sensor and Robot Design In order to identify *safe-for-motion* regions, the sensor must provide a quality measure, which allows to distinguish measurement-wise (per sensed point), whether the ray is free of obstacles or not. A long measurement distance d_{max} , e.g. in the range of the link length of the robot, is beneficial, yet useless without the quality measure. The prioritized feature in sensor-based exploration is the quality measure, as the expected results correspond well to the actual measurements. If 2-D sensors are applied, the robot system must allow for an orthogonal scan and planning joints. A rotational degree around the last axis improves performance only if a 2-D sensor is mounted centrically. For stereo 2.5-D sensors, rotation in sensing direction is optimal, otherwise, a sensor movement perpendicular to the scanning joint is recommended.

Sampling Technique The sampling technique must deliver a sparse map for efficient motion planning and path finding. The Dijkstra (cf. Sec. 2.4.4) method always considers the entire roadmap, thus, its efficiency highly depends on the roadmap size. Sampling a visibility-based roadmap with a $\mathcal{P}$-boundary constraint optimally provides possible view nodes.

Update Rule Fuzzy and Bayes' Update perform similarly w.r.t. the exploration results. While Fuzzy requires no independence of cells and measurements, its update is conservative. The limitation on the grey value of a cell's state, computed from two independent sets, reduces the potential of Fuzzy. Therefore, the application of Fuzzy does not improve the performance in comparison to Bayes'. The grey scale value,

i.e. cell state, inherits another challenges when applying Fuzzy Update: The sets $\mu_{free}(x)$ and $\mu_{occ}(x)$ consistently grow with every measurement; once a region is designated *free*, it can never be denoted *occupied* again. As the Belief Update is slow and does not show a notable advantage in comparison to Bayes', the Bayes' Update Method is selected.

Planning and Sensing Strategy Multisensory measurements provide an optimal exploration, provided that the sensors are selected properly. Each sensor compensates the weaknesses of its counterparts. The planning should be performed with the *safe-for-motion* sensor, as it definitely affects the information gain, whether sensing an obstacle or not.

This sensor should be combined with a long-range sensor, which must not necessarily be *safe-for-motion*. The measurements of this sensor definitely increase knowledge.

The lower bound of the IG is defined by a third sensor contributing a quality measure. If the sensor FoV is aligned with the rotation axis of the robot TCP, a rotational scan, i.e. $q_{scan} = 6$, should always be performed. Otherwise, a sweep with the second last joint of the Kr16, i.e. $q_{scan} = 5$, is recommended. The PBV strategy is optimal in most cases. BRS is preferable in highly occluded environments. The sweep joint must be selected considering q_{scan}.

Following these considerations, the optimal sensor measures 2.5-D information, assigning a quality value per individual measurement. As this is generally not feasible in a single sensor, a combination of at least two sensing principles is recommended. The *safe-for-motion* sensor can be short-range when combined with a long-range sensor without a quality feature. In order to increase the robustness of the operations, a third sensing principle is beneficial.

For this combination of sensors, an application of Bayes' Update performs best. The ideal planning strategy is PBS, which optimally combines direct and randomly sampled computation of NBVs. Set A displayed in Tab. 6.9 represents the optimal combination of sweep and scan joints for standard industrial robots with three intersecting axes in the wrist.

7

Multisensory Eye-in-Hand System

This chapter presents the design and implementation of a multipurpose vision platform suitable for exploration, inspection, and object recognition in sensor-based autonomous operation.

Numerous robotic applications require perception, yet to date, a sensor system applicable in versatile fields such as object recognition, cultural heritage preservation, or visual servoing is missing. While in industrial robotics the nominal model is known in advance, in small batch assembly as well as in service robotics the environment of a robot is not pre-defined. Therefore, a highly flexible sensing system is required, motivating the development of such a sensor rather than a perfectly specialized system. Flexibility in measurement range, sensing principle and processing is desired; further, the data and control interfaces of the system should be generic.

In this thesis, the focus is set on multisensory exploration. The methods developed in the previous chapters require certain sensor features in order to generate *safe-for-motion* areas.

In the following, design principles for such a flexible, multisensory vision platform, as well as their implementation in the DLR *3D-Modeller* are described. In specifying and combining multiple sensors, i.e. laser-range scanner, laser-stripe profiler and stereo vision, the mechanical and electrical hardware design for an integrated system is derived. Reliable concepts for synchronization and communication complete the approach. The versatility of the 3D-Modeller is illustrated by addressing four applications: 3-D modeling, exploration, tracking, and object recognition. Due to its low weight and the generic mechanical interface, this sensor can be mounted on industrial robots or humanoids, or can be used free-handedly. The 3D-Modeller is flexibly applicable,

not only in research but also in industry, particularly in small batch assembly.

In this chapter, the design criteria for the sensor development are elaborated, leading to the concept for designing a robotic system. Further, the 3D-Modeller system is presented in detail. The section concludes with an elaboration of possible applications of the system.

7.1 Design Criteria

Current commercial and research vision systems are usually specifically designed for one or very few applications and are therefore not suitable for flexible use in applied research or in SMEs: Various applications (hand-guided digitization, robot vision, automatic robot work space exploration [159], and more see Sec. 7.3) have different and sometimes conflicting demands concerning sensor range, view angle, lighting, precision, and acquisition speed. Furthermore, there is a need for a compact, versatile mechanical platform e.g. for the use of different pose reference sensors, a handle for hand-guided 3D-modeling, or interfaces for robot applications.

These demands can only be met by combining several sensors on the same hardware platform.

The 3D-Modeller design enables a multiple and flexible use in various applications. Furthermore, the system supports different operation modes (hand-guided and actuated, i.e. robot-controlled) and different pose reference systems, demanding a modular framework.

In the following, the sensors integrated in the 3D-Modeller are briefly described. Then, requirements derived from the envisaged applications are presented.

7.1.1 Sensor Types

The implementations of the abstract sensor types introduced in the previous chapters, i.e. LRS, LSP, and SCS, are detailed and specified in the following. The simulation results concerning the influence of the sensor type on the exploration performance (cf. Sec. 6.3.3) are the basis for an efficient combination into a comprehensive multisensory system, applicable for the tasks specified in Chap. 5.

DLR Laser-Range Scanner

The DLR Laser-Range Scanner (LRS) [61] uses the principle of laser triangulation. The outgoing beam is generated by a laser diode (670 nm) and focused by a highly refracting micro lens. The transmitted light is dispersed diffusely on the object surface. Some of the reflected rays

strike the receiver lens, which focuses the light onto a Position Sensitive Detector (PSD). A special challenge for this sensor are multiple object surfaces with widely different reflection characteristics. These necessitate a high dynamic adaptability of the sensor. For this reason, the transmitted power is automatically adapted to the optical characteristics of the measured surface in every single measurement. As the laser-range scanner rotates around its longitudinal axis, it can quickly gather a cross-section of distance values in a range of 270 degrees. Because of its robust 2-D data acquisition and the small dimensions, the LRS is used as a high-definition short-range sensor. Its wide scan angle has notable advantages in robot vision.

Stereo Vision

Stereo Vision is very efficient for sensing textured surfaces. A stereo sensor acquires large areas of the environment at once. The accuracy is usually low, but the sensors cover a wide range. The base difference of the cameras defines the minimum and maximum sensing range. It is desired to cover a range of approximately $250\ mm$ up to $2000\ mm$. Optical filters (as normally required by laser-stripe profilers) are not applied, as they would make stereo processing impossible. In this system, innovative stereo processing as presented in [70], [71], and [68] is implemented. This achieves a high ranking in comparison to other stereo methods [82], which are evaluated by Scharstein et al. [132].

Single and Dual Laser Stripe Profiler

The single Laser Stripe Profiler (LSP) consists of a laser beam that illuminates a stripe on a surface, and a camera that records its reflection. The 3-D position of different points contained both in the stripe and on the surface may be readily estimated by means of image processing algorithms followed by triangulation – provided a calibrated system [151] is used.

Scanning with this kind of sensor is almost like virtual spray-painting, particularly if the sensor is mounted on a hand-held device, as it can be swept freely over an object's surface. However, gaining data while horizontally moving in stripe direction is not possible. During automatic scanning using a robotic manipulator, this fact constrains the robot's movement and results in the loss of one DoF.

In order to eliminate this constraint, an additional laser beam that illuminates perpendicularly to the first stripe is used. Due to construction-related constraints, both laser beams have to be placed in close vicinity of each other. Since this arrangement causes an undesired reduction of the base distances between each laser beam and the camera, the second camera integrated in the 3D-Modeller is used.

Each of the cameras performs single LSP scanning with the laser stripe farther from it (cf. Fig. 7.7(a)). This configuration is denoted Dual LSP.

The addition of a second miniaturized laser beam results in:

- the release of a scanning movement constraint;
- the increase of surface-related information gained in every direction;
- the possibility to duplicate the sensing rate, as both cameras and laser beams can be triggered in a complementary way at highest speed and limited shutter time.

Robust image processing supports the segmentation of the laser-stripe without the need of optical filtering. The LSP delivers close to mid-range distance measurements, yet no quality value is provided.

7.1.2 Specifications

A light-weight design of the 3D-Modeller is required, as the system must be usable in hand-held operation. Therefore, the overall weight of the system must not exceed $1\,\mathrm{kg}$. The center of mass location is to support ergonomics. In the following, further requirements for different work space applications are presented and summarized.

Requirements for Exploration

Exploration requires sensors that detect *safe-for-motion* areas. Therefore, a quality measure which confirms the reliability of the sensing results is required. When mounted in hand-eye configuration, the sensors should have a large opening angle in order to provide good vision. Particularly when applying the beam planning method described in Sec. 3.2.2, the measurement range should ideally be the link length between sensor joints and robot joints, e.g. for a Kuka KR16 the length $l34 \approx 400mm$, cf. Tab.6.2.

The LSP applies the same laser intensity for the entire line, therefore the measurement $d_i = 0$ (the laser line is represented as n discrete points) can either mean that the entire sensing range is free, or that an obstacle's surface has absorbed the beam. In Fig. 7.1, such a situation is displayed. However, if the LSP detects an obstacle, the measurement is quite accurate and the region can be sensed to be free rapidly due to the large measurement range.

While for modeling good- and high-quality range points are segmented, the low reflectance values are not segmented, resulting in $d_i = 0$. The interpretation of these values is not possible, reducing the exploration

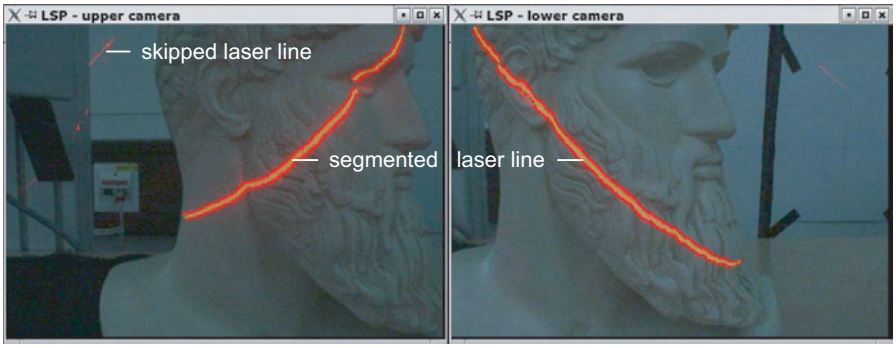
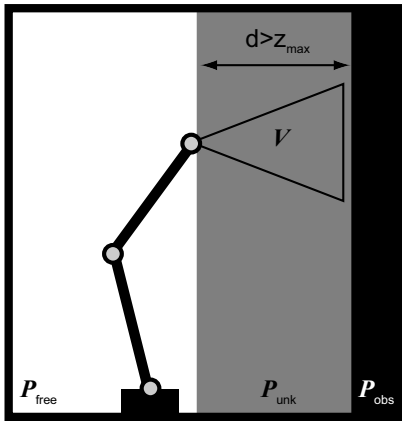
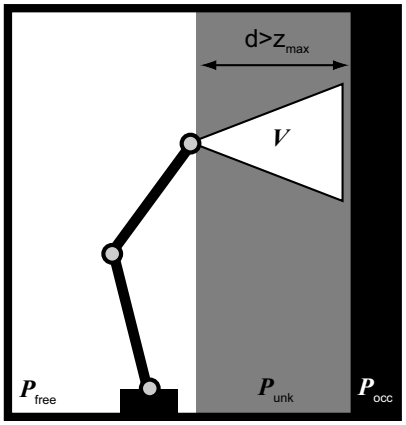


Figure 7.1: Camera images of the dual LSP with segmented laser-line. In the image on the left, the laser line is visible but not segmented.



(a) Sensor without quality measure



(b) Sensor with quality measure

Figure 7.2: *Safe-for-motion* part of physical space, i.e. $\mathcal{P}_{free}$, expansion (idealized) after scanning with a sensor for modeling (left) and a sensor for exploration (right).

performance. The LRS measures point-wise; the laser intensity is increased until a pre-defined threshold is passed on the receiver. This laser intensity is used as a quality measure. If the intensity is high, yet no obstacle is detected, the assumption that no obstacle is present in the sensing direction is justified, i.e. $d_i = d_{max}$. This results in an enlargement of $\mathcal{P}_{free}$ as depicted in Fig. 7.2.

While a stereo sensor has the largest FoV, it is prone to the same systematic error as the LSP: Its passive measurement technique cannot guarantee an obstacle-free ray in case nothing is measured. Furthermore, textured surfaces are required. However, the Semi-Globale Matching (SGM) method [70] is useful, as mutual information delivers more measurements on plane surfaces than pure correlation-based methods [82].

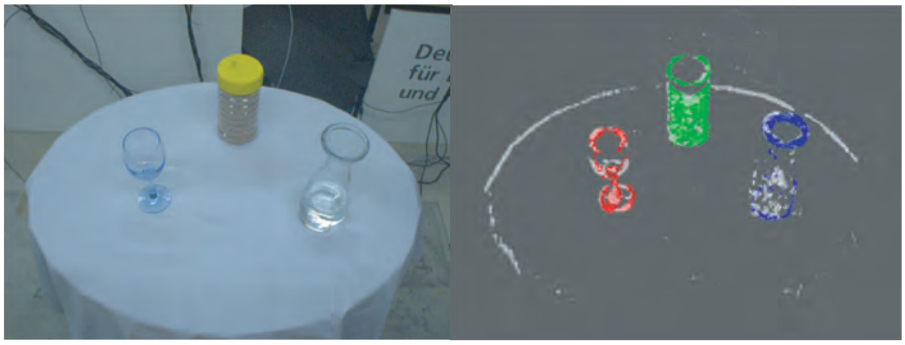


Figure 7.3: Table top scene with glass, tea box, and jug (left). The light data points are obtained using the MWMF method, fitted with model data points obtained using the identical method [121].

Requirements for Object Recognition

For object recognition and pose estimation [10], a large FoV is beneficial. In most cases, many views of an object are aligned, requiring 2.5-D images. While laser scanning delivers high-quality surface models, recognition focuses on features which are unambiguous for the object. Some methods work on surface models [174], others on point features [66]; especially for pose estimation, a fast method is required. Correlation-based stereo methods like the MWMF [71] produce good results, even for objects difficult to acquire, e.g. glassy surfaces. An example of a typical situation is depicted in Fig. 7.3, obtained with the 3D-Modeller on Justin (cf. Fig. 7.11).

Requirements for Surface Modeling

For surface modeling, noiseless measurements with high preciseness are required. Each noisy measurement tends to cause problems in estimating the surface normal. This leads to an enlargement of the minimal edge length [19], which smoothes the resulting model. In this application, good range points are required. Holes caused by uncertain measurements are closed in post-processing, e.g. by application of a hole filling algorithm [97], or by re-scanning the region. In hand-guided operation, a re-scan is a cheap operation, whereas in automatic modeling it implies high motion and sensing cost. Therefore, bad points are usually filtered out in surface modeling, whereas in exploration these points are very valuable for determining obstacles with non-ideal surfaces.

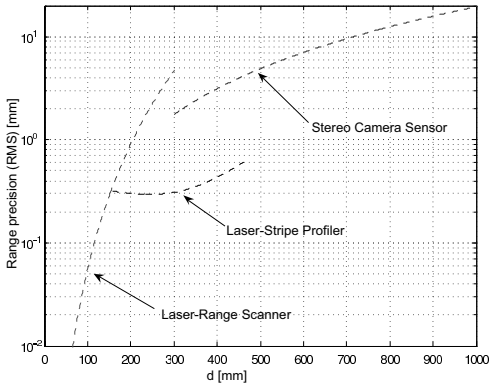


Figure 7.4: Range precision of the different sensors. The LRS has the smallest range but highest precision, making it suitable for hand-guided 3-D modeling. The LSP covers the medium-range, most applicable for exploration and 3-D modeling. Stereo ranges up to 2m, its optimal application is object recognition due to the large FoV (see Fig. 7.7(b)).

Table 7.1: Technical Data of the 3D-Modeller's sensor components. Two options for lenses are given, differing in focal length f, i.e. f_1=6mm or f_2=12mm

Sensor	Laser-range scanner	Dual LSP	Stereo Camera Sensor
Principle	triangulation	image processing and triangulation	stereo vision
Range [mm]	$50-300$	$150-500$	$250-2000$
Resolution [mm]	$0.1-2.5$	$0.3-2.5$	$2-50$
Base distance [mm]	20	103	50
FoV	270°	58° (f_1)	58° x 44° (f_1)
		30° (f_2)	30° x 22° (f_2)

Requirements for Tracking

For tracking objects in 6-D [138], high frame rates are desired. It is not tractable to compute range points; instead, a previously generated textured 3-D surface model is tracked in 2-D images in video frame rate. In this application, camera images are required; in case of visual servoing these must be synchronized with the robot that the cameras are mounted on. With an increasing frame rate in stereo processing and the development of 2.5-D range cameras, direct work in these range data sets in real-time is likely to become feasible.

Conclusion

Design considerations led to the technical specifications for the individual sensors integrated in the 3D-Modeller. Fig. 7.4 quantifies the range precision of the sensors. Tab. 7.1 summarizes the measurement ranges as well as the FoV. The stereo method must be exchangeable,

modeling and exploration require SGM, whereas object recognition is more robust on correlation-based methods delivering only prominent features such as edges.

While active systems are more robust, the effort for illumination increases massively with the size of the field of view.

$$p(D = d|d_{obs}) = \frac{1}{\sqrt{2\pi}\sigma_d(d_{obs})\sigma_\alpha} e^{-\frac{1}{2}(\frac{(d-d_{obs})^2}{\sigma_d^2(d_{obs})}+\frac{\alpha^2}{\sigma_\alpha^2})} \tag{7.1}$$

The variable d denotes the measurement, given an obstacle at distance d_{obs} under the angle α. The variable σ_d characterizes the distance variance; σ_α describes the angular variance of the sensor. A typical characteristic for the variance is an exponential function, which increases with a growing distance:

$$\sigma_d(d) = k \cdot d^i \; ; i, k \in \mathbb{R} \; . \tag{7.2}$$

Parametrical formulations are also possible. Additionally, no angular variance is taken into account, even though the sensor has an angular discretization of $\triangle\alpha_{res} = 0.9°$. The omission of the angular variance is justified by the high angular preciseness of the laser-based sensors as opposed to sonar sensors, where a consideration of σ_α is required, leading to $\alpha = 0$. The simplified sensing model can be denoted as follows:

$$p(d|d_{obs}) = \frac{1}{\sqrt{2\pi}\sigma_d(d_{obs})} e^{-\frac{1}{2}(\frac{(d-d_{obs})^2}{\sigma_d^2(d_{obs})})} \tag{7.3}$$

The characteristics of the precision are exemplarily presented in Fig. 7.4

The LSP's precision can be computed very similar. For this sensor principle, it is important that even though rectified camera images are used, the accuracy does not only depend on the distance, but also on the angle relative to the center of the camera, requiring an application of the general model from Eq. (7.1).

However, for view planning and measurement integration with LSP and SCS only the range precision is considered in applying the beam-based model.

The properties of the sensor are visualized in Fig. 7.5. They are described in the local sensor coordinate system F_{sensor}. The variable ω denotes the angular velocity of the laser-range scanner (cf. [154]), while α_{on} and α_{off} denote the angles at which the laser is turned on, respectively turned off, limiting the FoV. The maximum sensing distance is denoted d_{max}, the minimum distance d_{min} is not considered in the following calculations. The actual distance d equals the mean μ of the Gaussian.

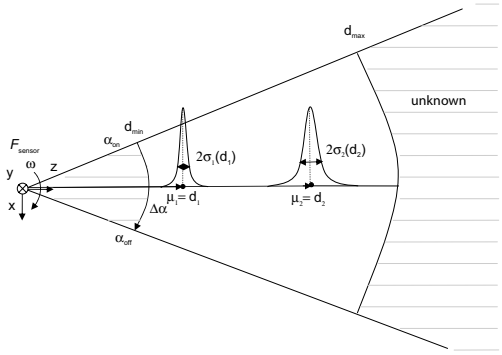


Figure 7.5: Properties of a general 2-D sensor. The variance σ increases with the measurement range. The sensing range is bounded by a minimum and maximum sensing distance. Obstacles can only be measured in $d_{min} < d < d_{max}$. The opening angle $\alpha \in [\alpha_{on}, \alpha_{off}]$ is displayed for a 2-D FoV.

The following section presents the robotic version of the 3D-Modeller, designed and implemented based on the experiences and results gained with the hand-guided system (cf. Chap. C). The system is developed according to the design rules formulated in Sec. 7.1.

7.2 Robotic Multisensory 3D-Modeller

Perception is the process of acquiring, interpreting, selecting, and organizing sensory information while yielding internal representations of the world. In robotics, there has always been a high need for perception of the environment, especially with the purpose of increasing the level of autonomy of robotic systems.

Currently, there is a strong need for perception systems that are applicable for flexible work cells in the industrial robotics domain. The pose and models of objects are to be estimated (inspection), further, unknown objects are to be detected and autonomously identified (exploration). This plethora of different functions calls for the development of a highly integrated 3-D sensor device. The development of a robust, flexible 3D-Modeller is of paramount importance to current technological challenges such as automation in small batch assembly (for the reduced setup costs of integrated sensing) or cultural heritage preservation (for the convenient handling and the simultaneous gathering of texture and geometric structure).

The requirements for a multi-purpose vision platform [158] can be summarized as:

- complementary range sensing principles that allow data fusion;
- extendibility/flexibility in sensing;
- synchronization through adaptable measurement clocks;

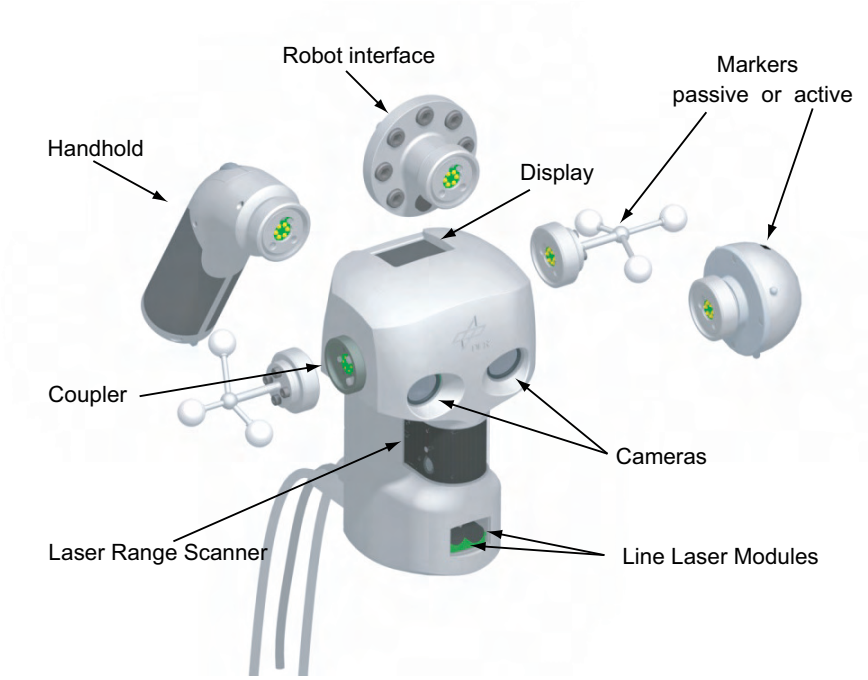


Figure 7.6: 3D-Modeller components.

- generic mechanical interfaces;
- compact, light-weight design;
- local computational capabilities (smart sensor);
- extensive software suite.

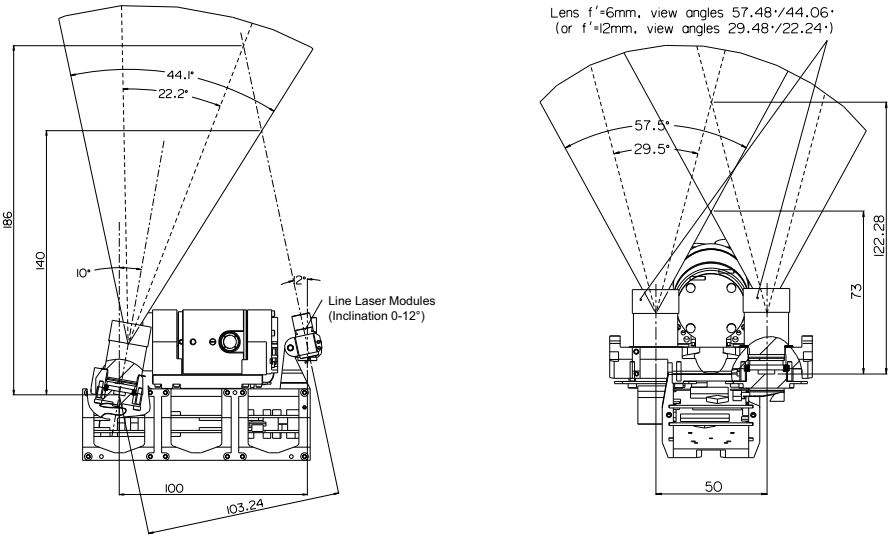
The consideration and implementation of the requirements is detailed in the following sections.

7.2.1 Hardware Implementation

In this section, the hardware components and the corresponding technical data are described. The processing system architecture is shown in Fig. 7.8. The design considerations from Sec. 7.1 lead to the following mechanic and electronic hardware layouts:

Mechanics

The 3D-Modeller is designed to arrange all components as compactly as possible, ensuring good usability and robustness at the same time. The total weight of the system is $850\,\mathrm{g}$, making it suitable for hand-held operation.



(a) side view: laser-stripe profiler

(b) top view: stereo camera sensor

Figure 7.7: 3D-Modeller: FoV of the optical components.

It is equipped with three identical easy-to-connect free-of-play couplers. They contain six electrical contacts for active markers[1] and the handle. The housing is coated EMC-compatibly. The 3D-Modeller consists of the following components:

Cameras: Two FireWire cameras from AVT[2] are integrated for stereo vision algorithms [70] and texture mapping. In combination with the line-laser modules, they form the LSP [151]. Following the compact design, the cameras are embedded in the 3D-Modeller without their original housing and with customized lens holders for off-the-shelf lenses[3]. There are two different adaptable lenses (f_1=6mm or f_2=12mm). The base distance between the cameras is 50mm, derived from baseline optimizations, which is a trade-off of between perspective difference and range precision.

Laser modules: For the LSP, two line-laser modules with 60 degree opening angles are used. Due to optimal camera sensitivity, the wavelength of the laser diodes is 635nm. Space restrictions require the laser modules to be a proprietary development. The laser modules are individually inclinable for different measurement ranges (Fig. 7.7).

[1] Optical tracking system supports passive retro-reflective markers as well as infrared emitting diodes, so-called active markers

[2] Allied Vision Technologies GmbH, http://www.alliedvisiontec.com Link: 2007-08-01

[3] For use with Sony NF-mount objectives

Handhold: The handhold is designed for easy use and good ergonomics in hand-held operation. It contains buttons and scrolling wheel of an off-the-shelf USB computer mouse. The handhold is hot-pluggable to the embedded PC Board (see Sec. 7.2.1) and its operation is user-friendly.

Tracking Markers: Optionally, passive or active markers for tracking can be coupled to the system. Exchanging them does not demand recalibration because free-of-play couplers are used. The arrangement of the markers is optimized w.r.t. marker visibility and tracking accuracy.

Mechanical Connector: The mechanical coupling on the rear side of the 3D-Modeller is suitable to plug the handhold as well as to connect it to a robotic manipulator, allowing the system to be "plug-'n'-play"-able.

Electronics: The concept of the 3D-Modeller electronics reflects the intention of reducing the amount of required cables. The system is designed as a smart sensor (see Fig. 7.8), yet not all processing can be done locally in the 3D-Modeller, therefore an external sensor-PC is needed for computation. Cabling between these systems has been minimized. FireWire was chosen as a suitable bus medium for this purpose, as it allows both communication and power transmission.

Embedded PC and Peripherial Connection: The embedded PC[4] is the core element of the electronic concept. In addition to SDRAM, the module is equipped with flash memory to locally boot from and store the operating system and applications. The embedded PC controls the synchronous cooperation of all sensor components and manages the communication with the sensor-PC. The embedded PC is mounted on a mainboard which carries all electronic parts needed to connect to peripheral components and interfaces. Further, the mainboard provides the electrical power for the embedded PC and the peripherals. It is designed in flex-rigid technology, thus it can be folded into the housing of the 3D-Modeller and connects all components without any jacks.

IEEE1394 Interface: A IEEE1394b[5] 3-port physical layer controller and a link layer controller are implemented on the mainboard. A chipset from Texas Instruments is used, wired to the embedded PC through its PCI-Interface. The IEEE1394 Interface has several duties:

[4]Kontron X-board PXA, http://www.kontron.com

[5]FireWire and IEEE1394 are concurrently used throughout this paper. The appendix b denotes the standard allowing for data transfer up to 800Mb/s

Two of the FireWire ports connect the IEEE1394a-cameras to the system. The communication with the sensor-PC is done via the remaining FireWire IEEE1394b port.

Synchronization Interface: All components of the 3D-Modeller must work synchronously in order to allow sensor data fusion. For this purpose, a synchronization unit is implemented on the mainboard.

Laser Module Drivers: The two line laser modules consist of laser diodes which can be operated either continuously or pulsed. In pulsed operation mode, the two laser modules are toggled by the video frame clock. Thus, the laser stripe can be separated by comparing subsequent video frames. The clock synchronization and the power control of the line laser modules are implemented on the mainboard.

Interface to Active Markers: The 3D-Modeller can optionally be equipped with active markers that emit infrared light, detected by tracking cameras of the 6-D pose reference system. The mainboard provides the clock signal and power needed for the active markers.

User Interface: The user interface consists of an 1.8" color TFT display with a resolution of 160x128 pixels, directly controlled by the XScale on-chip LCD controller and an USB root hub. For user input, a mouse-like button and scroll wheel device, embedded in the handhold, is connected to this USB port when in hand-guided mode. Further, a USB hub for easy connection to other USB devices - e.g. a keyboard or a mouse - is integrated in the handhold.

Power Supply: All components of the 3D-Modeller, with exception of the laser-range scanner, are powered through the IEEE1394 interface by the sensor-PC. As the supply voltage of the IEEE1394 interface can vary in a wide range, the mainboard provides voltage control for all required supply voltages of the embedded PC and its components. In order to achieve this with a minimum of waste heat, highly efficient switching controllers are used.

Laser-Range Scanner: Due to its complexity, the laser-range scanner is the only component of the 3D-Modeller that is not yet fully integrated into the IEEE1394 communication concept. For historic reasons, the LRS transfers its data to the sensor-PC directly via a CAN-Interface (cf. Chap. C). Nevertheless, it is integrated into the global synchronization concept through a separate clock wire.

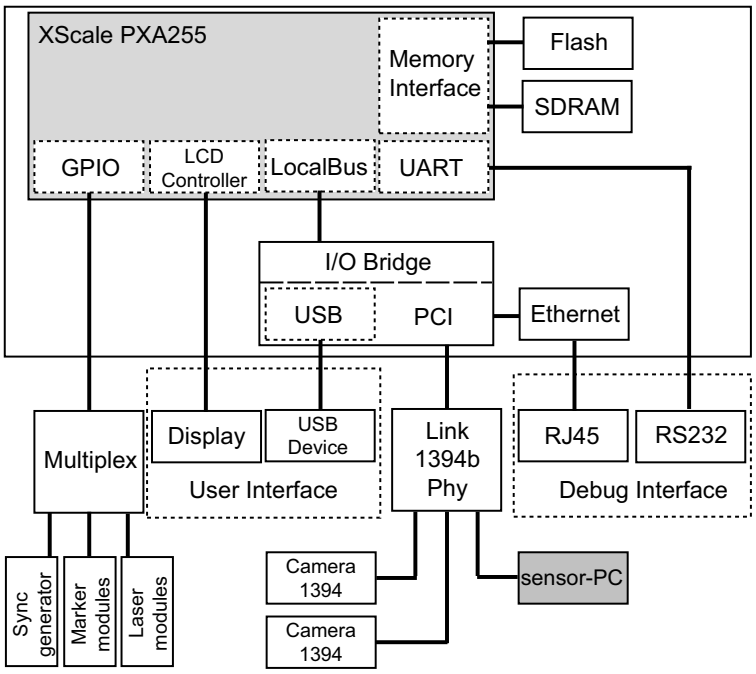


Figure 7.8: Communication concept.

7.2.2 Communication

The communication concept of the 3D-Modeller integrates the different control tasks. These comprise the control of the laser modules, the active markers, and the synchronization signal generator, as well as the management of an LCD display, an input device, and the establishment of an interconnection between 3D-Modeller and sensor-PC.

In normal operation mode, FireWire for both data transfer and power supply is the only connection between the device and the sensor-PC. Using FireWire allows the power supply of the device via the same standardized cable and connector as the data flow. Further, the specification of FireWire as a data link permits a straightforward integration of the two FireWire cameras, communicating via IIDC protocol to the sensor-PC's software suite. A design guideline for the specified processing system is to use standard hardware and software components wherever applicable in order to manage complexity and reduce design cycle time.

The selected operating system is a Linux standard distribution for ARM processors[6]. Its operation merely requires minor adaptions to the

[6]The ARM architecture (previously, the Advanced RISC Machine, and prior to that Acorn RISC Machine) is a 32-bit RISC processor architecture developed by ARM Limited that is widely used in a number of embedded designs. Because of their power saving features, ARM CPUs are dominant in the mobile electronics market, where low power consumption is a critical design goal.

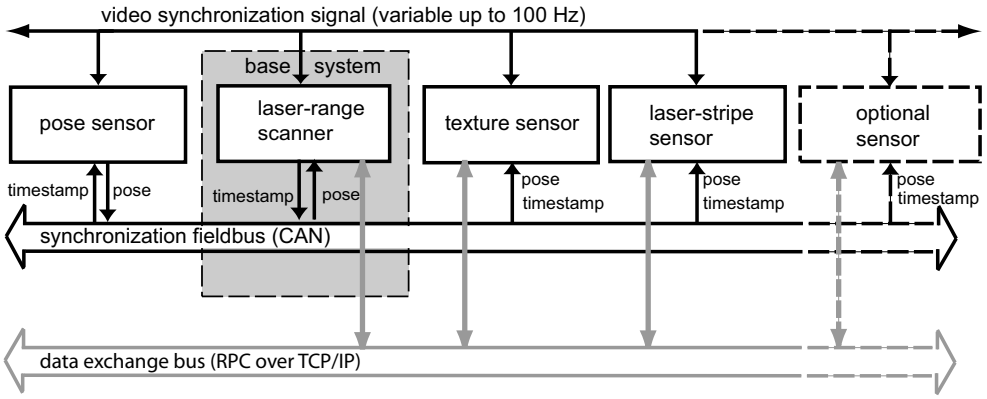


Figure 7.9: Extended synchronization concept for the robotic 3D-Modeller: hardware level (video), software level (CAN), data exchange level (RPC).

standard configuration, as well as the implementation of the device drivers for the sensor interface hardware. The Linux operating system is bootable from the flash memory and needs no connection to a host computer for start-up. Once running, it uses the IP over FireWire protocol to establish a standard network connection to the sensor-PC.

The stringent adherence to standard software components enables access to well-known libraries, interfaces, and services, e.g. Qt embedded for graphical user interface, RPC-based communication services, and simple shifts of functionality from sensor-PC to the 3D-Modeller on board. The communication concept supports an easy implementation as well as the adaption of required and designated software functionality and represents a rapid prototyping platform for the embedded software.

7.2.3 Hardware Synchronization

Hardware synchronization is crucial for systems composed of more than one sensor. All sensors acquire data simultaneously, making e.g. fusion of pose and image data possible. For the previous version of the 3D-Modeller (cf. Chap. C), the hardware synchronization signal is done via analog video cameras, which provide a clock signal. Digital FireWire cameras do usually not provide a hardware signal, but can be externally triggered. Therefore, a flexible strategy to supply all sensors with a common clock signal is implemented in the current system. The synchronization signal can be changed as depicted in Fig. 7.9.

The synchronization unit depicted in Fig. 7.8 can work in two different ways: Either it generates a video clock signal internally and provides it

http://en.wikipedia.org/wiki/ARM_architecture LINK: 2007-08-17

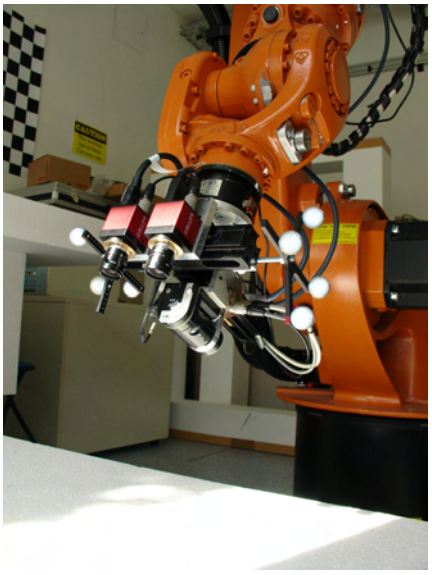


Figure 7.10: 3D-Modeller components mounted on a robot: All sensors are attached to optical rails to optimize the base distance. This 3D-Modeller variant is used for evaluation of additional sensors. In addition, sensor comparison studies can be performed, further, localization methods [34] can be tested.

Figure 7.11: 3D-Modeller implemented as vision system of the DLR humanoid robot *JUSTIN*: The system is actuated with a two DoF pan-tilt unit to allow head movements. Two-handed manipulation of known objects, after determining their pose with the 3D-Modeller, is demonstrated.

to all connected components, the timing parameters of the video sync signal being freely programmable by software through the embedded PC. Alternatively, the synchronization unit receives an external video signal from another source (e.g. an analog camera or an external pose sensor) and synchronizes the cameras and line laser modules on this external clock.

7.3 Applications

In this section, the versatility of the 3D-Modeller is applied to solve perception tasks in robotics. Primarily designed for exploration, it can be used for further applications, making it a notable first step towards a sensor extending the autonomy of robotic systems in unstructured environments. First, applications in research are presented. Furthermore, the use envisaged in industry, e.g. for automating processes with small batches, is addressed.

7.3.1 Applications in Research

The following applications are merely examples of a wide range of possibilities in research:

- online surface triangulation [19];
- robot work cell and configuration space exploration [159];
- texture-based real-time tracking of objects [137, 138];
- vision system for object recognition for two-handed manipulation with the humanoid robot system JUSTIN [121];
- modeling in cultural heritage [72, 104].

The latter is described in further detail in the appendix Sec. E.2, as first implementation steps have already been performed, showing the potential reaching beyond robotics.

Applications discussed in Sec. 7.1 have been successfully tested in the past. However, most tasks were performed in manual operation mode or with complete knowledge of the scene w.r.t. maneuverability of the robot. The online surface acquisition task was performed manually (also considering the case when the system is mounted on a robot which is jogged manually). The object recognition task for JUSTIN is implemented as *look-and-move-approach*, exploration of the object and scene is not considered so far. The tracking requires knowledge on the free space around the robot. To date, the described approaches use the 3D-Modeller, however, no environment model has been computed to constrain the motion planning yet.

The scope of applications validated in this thesis focuses on the exploration problem, however the exploration task is integrated in a framework for combining exploration, inspection, and object recognition for use in service robotics and flexible work cells. This framework is described in Chap. 5, an overview of the tasks is provided in the respective section.

7.3.2 Application in Industry: Small and Medium-size Enterprises

The robotic 3D-Modeller was developed in an Integrated Project in the 6^{th} Framework Programme of the European Commission SMErobot™[7]. This project aims to provide methods and systems enabling SMEs to escape from the automation trap. Usually, automation of processes is considered when large lot sizes are manufactured, e.g. in the automotive industry. In SMEs, lot sizes are smaller, processes must be

[7] http://www.smerobot.org Link: 2007-05-01

flexible, and workers interact with the robot. In addition, spacious robot cells with safety fences are not affordable, as shop floor space is limited.

The multisensory approach with flexible couplers and extensive software for modeling, registration, and object recognition is beneficial to SMEs. The investment in a single sensor that can be applied to many tasks is preferable compared to the purchase of several specialized sensors, particularly when the requirements are not pre-definable. In order to prove the usability of the system in such environments, the 3D-Modeller has been evaluated by two leading robot manufacturers, Reis Robotics[8] and Kuka[9]. The applicability of the multisensory system is assessed within demonstrators developed by the two companies in the course of SMErobot™:

The application implemented and validated by Reis Robotics involves the installation of an industrial robot equipped with a novel set of tools in a joiners workshop.

Today, work pieces in such SME work environments are usually created based on crude hand drawings; oftentimes, no precise CAD models of the objects are available. An example of such a work piece, initially lacking a corresponding CAD model, is presented in Fig. 7.12. The ambition is to automatically perform the processes required for spray painting the work piece. As an initial step, the 3D-Modeller is installed on a Reis industrial robot and successfully applied to create an exact CAD-model of the wooden work piece, using its LSP sensor. This model is transferred to the Reis controller in a standard VRML2 format. Based on the surface model, the spray painting trajectories are planned by the controller and successfully performed by the robot.

Inspired by the wood-working-scenario, photographs taken in a joiner's workshop were used to design the cell partially depicted in Fig. 7.12. This setting is one of the two major test-beds within SMErobot™. Additionally, the experiments in Chap. 8 have been performed in this test-bed.

Further applications such as bin-picking using the 3D-Modeller's SCS are evaluated in cooperation with Kuka. In this application, the 3D-Modeller generates a point cloud, with is passed on to the object recognition method [66]. The model for this recognition is generated with the 3D-Modeller beforehand. Stereo shots from three different views are used to generate a point model of the objects to be recognized.

Further, the registration of work pieces for welding applications is evaluated, also in cooperation with Kuka. In this application, CAD models of the work pieces are available. A worker arranges the known CAD work pieces on a welding table. Then, fixtures are applied. In the manual work place in SMEs, the worker must perform a welding operation,

[8] http://www.reisrobotics.com/ Link: 2007-05-01
[9] http://www.kuka.com Link:2007-05-01

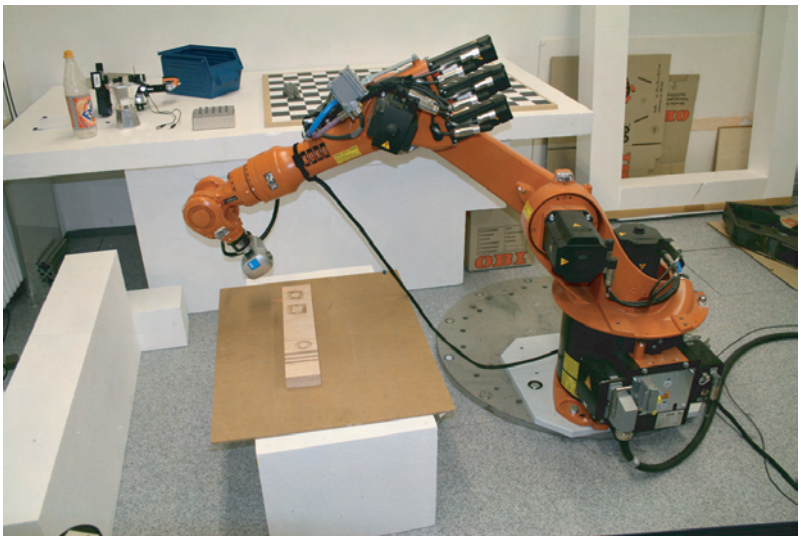


Figure 7.12: 3D-Modeller as vision system in a SME flexible work cell. The system is actuated with a 6-DoF Kuka KR16 in order to generate a surface model of a wooden work piece. Exploration and object recognition is performed in this test-bed, in addition the work cell is modeled for simulation purposes.

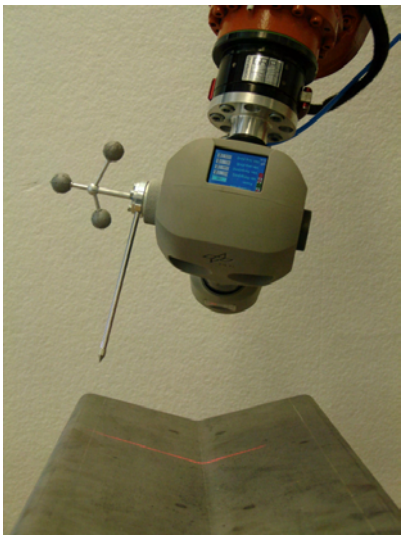


Figure 7.13: A metal work piece from the welding application is scanned with the LSP. Currently, these parts are fixed and welded manually. The task is to identify the pose of the parts in order to generate automatic welding paths.

which is repetitive. The 3D-Modeller, either mounted on the robot or operated manually, is supposed to measure points on the metal parts surface. These data is used to align the CAD model in the work cell coordinates, called registration.

7.4 Summary

The 3D-Modeller is applicable to various applications due to its combination of sensors. In this thesis, mainly the exploration and modeling capabilities are required. However, this system is designed to enable autonomous operation for multiple tasks according to the architecture depicted in Fig. 5.15. It is mountable to a wide range of robots. Thus, it is used for most 3-D vision applications at DLR, either to evaluate algorithms, or to be used in industrial applications and demonstrations.

8

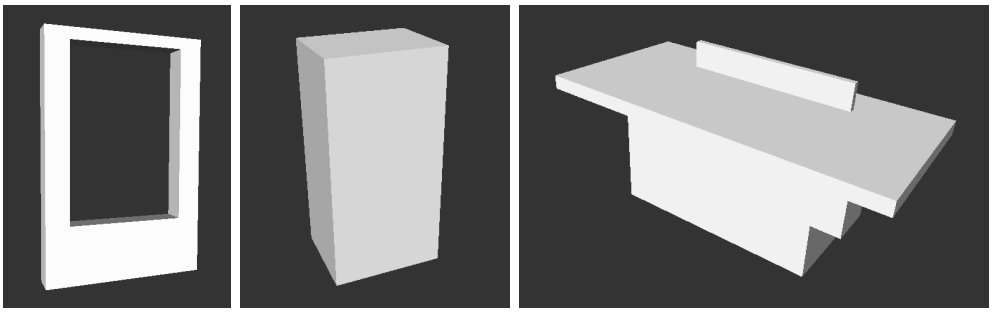
Experiments in Robot Work Cells

This chapter presents the exploration experiments autonomously performed by a robotic system in a realistic working environment. The experiments are supplemented by simulations in order to assess the differences between simulation and real life case. All experiments use a Kuka KR16 robot, an industrial robot with 6 active DoFs. The 3D-Modeller is mounted on this robot for multisensory measurements in hand-eye configuration.

The simulation tool SBP-Simulator is extended by interfaces to real sensors and the robot. $\mathcal{P}$-space representation, sensor models and update rules are identical to the simulations, enabling a seamless transfer of experience from simulation to reality. Further, simulations are used to predict exploration results. The parameters required for a successful exploration are determined in simulations and transferred directly to the experiment. This enables efficient preparation of the experiments in the laboratory, furthermore, simulations can be used in order to assess work cell layout parameters before setting up the real cell on the shop floor.

In this chapter, the overall integration of sensor, robot, and SBP-Simulator is described. The features of the real work cell for multiple task exploration are presented. Practical considerations for the experimental evaluation, e.g. joint limits, masking of measurements on the robot, and system calibration are elaborated.

Furthermore, the influence of the various update methods is evaluated in a series of experiments performing measurements on ideal and flawed surfaces with Bayes', Fuzzy, Belief, and Naïve Update in order to determine their impact on detecting *safe-for-motion* areas.



(a) Frame (b) Podium for objects (c) Work bench table

Figure 8.1: 3-D models of work cell objects used for the simulations and experiments in the test-bed. All objects are manufactured from polystyrene.

The results of multisensory exploration experiments in a flexible work cell are presented. Moreover, the automatic exploration of a RoI for modeling a Zeus bust is demonstrated. Finally, a multiple task exploration of RoI and $\mathcal{C}$-space is performed simultaneously. Conclusive discussions of the results close the chapter.

8.1 Test-Bed Setup

The test-bed consist of an industrial robot and the 3D-Modeller mounted on it.

Multiple polystyrene objects are arranged to imitate typical objects in flexible work cells. Additionally, other objects can be located unconstrained in the cell. A photograph of this test-bed is depicted in Fig. 8.3. In the following, the test-bed architecture and its components are described.

Further, the robot's properties are detailed, focusing on the joint limits used in the experiments and the relevant interfaces. Then, the relevant features of the 3D-Modeller's sensor components are detailed. The masking process for range measurements on the robot is elaborated.

8.1.1 Test-bed Architecture

The architecture of the applied components is depicted in Fig. 8.2. The 3D-Modeller is connected to a Linux PC. On this PC, the hardware abstraction layer program *SensorServer* is running. This software provides the 3D-Modeller's camera images, robot poses, and range data to the sensor programs for LRS, LSP, and SCS, which are running on the same PC. An event-based communication enables an efficient commanding of and data transfer from the sensors. Sensor range data

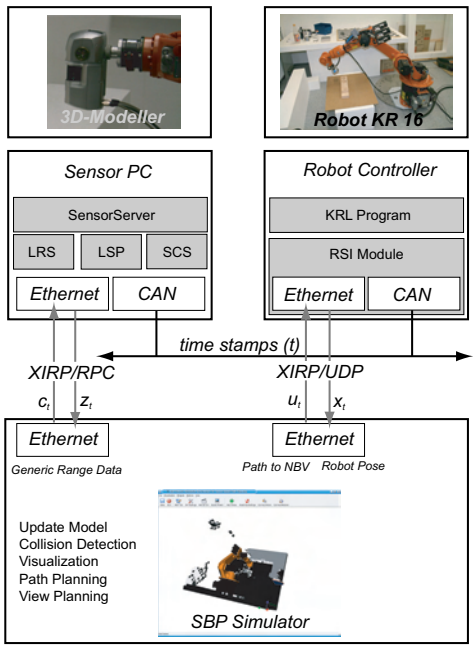


Figure 8.2: Architecture for interfacing 3D-Modeller, robot and simulation environment, i.e. SBP-Simulator.

Figure 8.3: Experimental test-best for sensor-based exploration.

is received via RPC by the generalized range interface[1]. The SBP-Simulator receives the sensor data and accumulates it in the sphere-tree implementation of the $\mathcal{P}$-space. The robot receives commands from the SBP-Simulator via User Datagramm Protocol (UDP) . A robot program[2] arbitrates the commands and transmits the current pose $\mathbf{q}_{curr}$, synchronized with the 3D-Modeller's time stamps. The current pose and its corresponding range data set are fused using time stamps every time a measurement is performed.

The robot system consists of a Kuka KR16 robot with a KRC2 controller and the TCP-mounted 3D-Modeller as depicted in Fig. 8.3. The 3D-Modeller's sensors are calibrated on the robot. The procedure is described in the appendix (cf. Sec. B.2). In the following section, the robot and its interface to the SBP-Simulator as depicted in Fig. 8.2 are detailed.

8.1.2 Robot

The experiments presented in this chapter are performed with a KR16 and the 3D-Modeller. The Kuka KR16 robot has 6 active DoFs and a payload of 16kg. In the experiments, the joint limits must be adapted to the test-bed properties (cf. Fig. 8.5). Tab. 8.1 shows the used joint limit set [T] in comparison to the joint limit set from the specification

[1] A XIRP interface is implemented, yet not used in the experiments
[2] Robot programs are usually implement in a high-level programming language

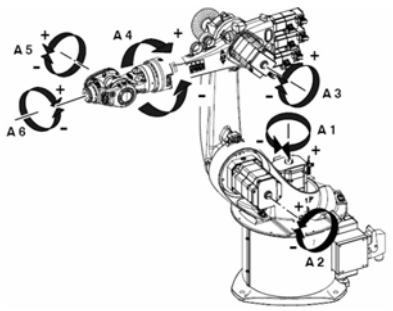


Figure 8.4: Kuka KR16 specification. (Source: http://www.kuka.com Link: 2007-07-25)

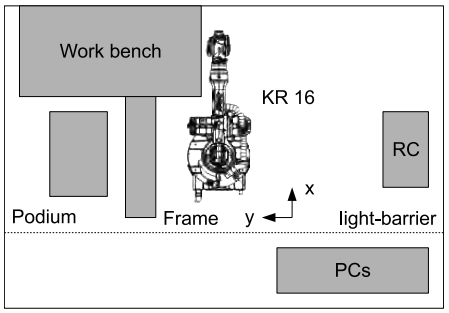


Figure 8.5: Layout of the test-bed. The robot moves in automatic mode, thus the access to the cell is secured by a light barrier. The space to the robot's left is used for the experiments. The base coordinate system of $\mathcal{P}$-space is depicted.

Table 8.1: Joint limits for the simulations with a Kuka KR16. Two sets are shown: [S] Kuka specification and [T] Test-bed specification.

Joint	Kuka specification [S]		Test-bed specification [T]	
	q_{min}	q_{max}	q_{min}	q_{max}
q_1	-185°	185°	-110°	0°
q_2	-155°	35°	-130°	-35°
q_3	-130°	154°	35°	135°
q_4	-350°	350°	-190°	190°
q_5	-130°	130°	-120°	120°
q_6	-350°	350°	-190°	190°

[S]. The limitations in joints q_6 and q_4 are required because of the 3D-Modeller's cabling restrictions; rotations of 350 degrees as defined in the specifications are prohibitive for these joints. The robot specification is depicted in Fig. 8.4[3].

Robot-Sensor-Interface for Synchronization The robot is integrated into the 3D-Modeller's synchronization concept as depicted in Fig. 7.9. The pose of the robot is coded with the 3D-Modeller's timestamp in order to fuse it with the corresponding range data sets. These timely coded robot poses are communicated via CAN bus.

Additionally, the SBP-Simulator must receive the current pose of the robot for two reasons: the robot kinematics module is updated with the current pose, and the sensings of the 3D-Modeller are appropriately integrated. This transfer is performed via UDP. Furthermore, the robot is commanded via UDP by the SBP-Simulator. Point-To-Point (PTP) commands are sent in order to guide the robot along the paths computed by the planning tool. The Ethernet and the CAN interface are implemented in a Robot-Sensor-Interface (RSI) module, running on the real-time part of the robot controller. In order to arbitrate

[3]The joints are denoted $A_i; i \in [1,6]$ instead of q_i, the notation used in throughout thesis.

the motion commands issued by the SBP-Simulator, a robot program in Kuka Robot Language (KRL) translates the commanded poses into robot commands. Poses can be commanded in joint angles or tool frame values. The communication is depicted in Fig. 8.2, a detailed description of the RSI module [43] is presented in Sec. D of the appendix.

8.1.3 Sensor System

The LRS, the single LSP, and the SCS of the 3D-Modeller (cf. Chap. 7) are used in the experiments. In the following, the sensor data interface is described. Then, the three sensor's relevant properties are summarized. Moreover, the sensor's combination in order to obtain optimal results in experiments is discussed. Concludingly, the masking of sensor measurements is elaborated.

Sensor Data Interface Exchangeability of robot and sensors is limited. For range sensors, a unified approach [20] is developed. The range sensors are classified by type; Cartesian, perspective, cylindrical and spherical sensors are differentiated. Each sensor provides a calibration matrix ${}_{TCP}\mathbf{T}^{S}$ which transforms sensor data into Cartesian TCP coordinates, and a transformation matrix ${}_{Base}\mathbf{T}^{TCP}$ which represents sensor data in the world coordinate frame. A time stamp and an array of depth values are transmitted. Optionally, a quality value per measurement, an indication for the reliability of the sensing, is provided. This interface is currently implemented for cylindrical and perspective sensors in the SBP-Simulator, covering the majority of sensors envisioned for exploration purposes. The generic range interface is implemented event-based, thus polling of the sensors is not required.

LRS A LRS measurement consists of a laser intensity in every point in addition to the distance value. This intensity is used as quality value $Q_i \in [0, 15]$, describing the laser intensity per measured point i. The quality value $Q_i = 15$ means that the laser emits at full power, yet no range measurement is obtained. In this case, the sensing beam is determined to be free of obstacles. The measured range values $d_m \in [d_{min}, d_{max}]$ are filtered based on the quality measure, d is processed by the sensor model.

$$d_i = \begin{cases} 0, & Q_i = \{0, 14\}; \\ d_{max} + 1, & Q_i = 15; \\ d_{i,m}, & \text{else.} \end{cases} \tag{8.1}$$

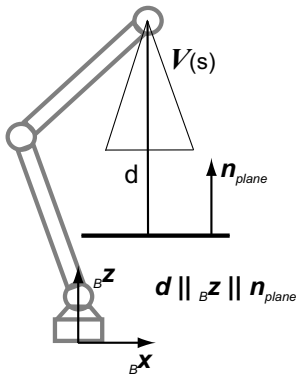


Figure 8.6: Set-up for measuring precision with the robot. The robot is moved vertically in parallel to the normal of the measurement plane. Distance sensings are performed, leading towards distance precision.

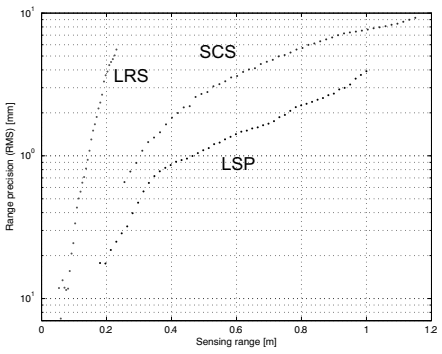


Figure 8.7: Measured range precision of the different sensors. The LRS has the smallest range but highest precision. The LSP covers the medium-range. The SCS ranges up to 2m with medium precision.

LSP The LSP is a conservative sensor; dark surfaces as well as specular surfaces do not allow reliable measurements. The laser intensity is not adapted individually but remains constant during data acquisition in the current implementation. Therefore, the $\mathcal{P}$-space is solely changed when an obstacle is measured. Otherwise, the state of the $\mathcal{P}$-space cells c_i $p^{k+1}(c_i) = p^k(c_i);\ \forall c_i \in \mathcal{V}$ remains unchanged after a measurement at time k.

SCS The SCS requires textured surfaces in order to operate properly. Its large field of view is beneficial, however, the sensing principle is not robust to illumination conditions due to its passivity. For exploration, the SGM-method [70] is advantageous to a correlation-based method such as MWMF [71], which is optimal for object recognition. Both methods are implemented for the SCS. The same restriction as presented for the LSP applies, nevertheless, the following binary quality filter is used:

$$d_i = \begin{cases} 0, & Q_i = 0; \\ d_{i,m}, & Q_i = 1. \end{cases} \tag{8.2}$$

The quality Q_i is derived from a successful match in the pre-defined disparity range (cf. [71, 70]).

Sensor Precision The parameters of range precision for LRS [79], LSP, and SCS are obtained by mounting the 3D-Modeller on the robot. A plane with normal $\mathbf{n}_{plane}$ in ${}_B z$ direction in robot base coordinates is measured. The TCP incrementally moves in positive z-direction in equidistant steps (cf. Fig. 8.6). The results are presented in Fig. 8.7.

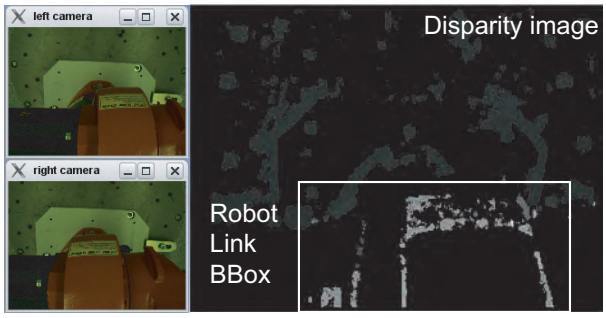


Figure 8.8: The robot scans itself with the SCS; as the MWMF [71] is used, mainly the robot edges seen in the disparity image are measured. An approximation for the AABB for link 3 is visualized.

Multisensory Data Acquisition In order to obtain *safe-for-motion* areas in physical space in combination with satisfactory exploration results, a multisensory strategy is performed. The suitable sensor for planning views must be selected. The LRS has the shortest range, yet senses regions in $\mathcal{P}$ most reliably. When planning with LRS, the IG after having performed the measurement corresponds to the expected values.

Due to the longer sensing range, planning with LSP or SCS has advantages for the beam model. Both sensors solely change the environment when measurements are obtained. In Point model, the $\mathcal{P}$-space block corresponding to the maximal IG location is not updated after the measurement if the entire FoV is free of obstacles, resulting in a unchanged IG after the sensing.

Fig. 8.10 shows the *safe-for-motion* area after planning a view with the LRS and executing a combined scan with LRS and LSP. The obstacle's distance from the sensor origin is larger than $d_{max,LSP}$. Therefore, views are planned conservatively, i.e. the LRS is applied for planning views, measurements are performed with LRS, LSP, and SCS, depending on the experiment.

Masking of Measurements on the Robot In the experiments, the sensor acquires sensings of the robot in certain poses (cf. Fig. 8.8), which results in occupied regions appearing directly on the robot links – the system stops moving as the robot is in collision (cf. Fig. 8.9). As the robot is not modeled in the $\mathcal{P}$-space sphere-tree, sensings of the robot itself do not occur in a simulation. In order to mask the self-scans in the experiments, measurements inside the bounding box of links 1 to 3 are rejected. The upper links cannot be measured due to the robot's kinematics, requiring no masking of measurements.

The following sections describe the results from the experiments, which are obtained in the test-bed.

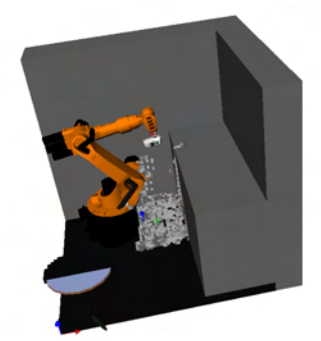


Figure 8.9: The robot scans itself with the SCS. No filter is applied, therefore measurements on the robot are interpreted as occupied in $\mathcal{P}$.

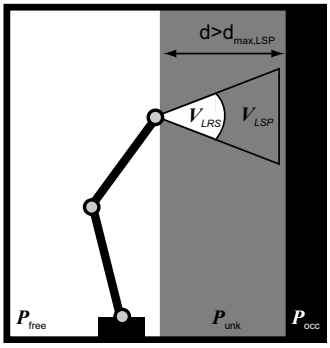


Figure 8.10: *Safe-for-motion* part of physical space, i.e. $\mathcal{P}_{free}$, expansion (idealized) after planning a view with LSP and scanning with LRS and LSP.

8.2 Experiments for Update Rules

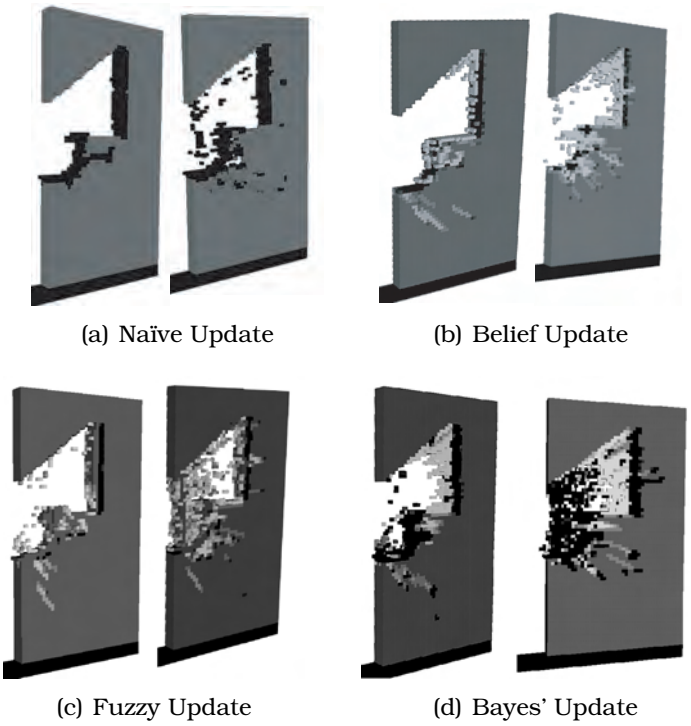
In this section, the influence of the surface properties is assessed. The interpretation of measurements is performed differently for each update type. The main purpose of the experiments in this section is the capabilities of determining *safe-for-motion* areas in $\mathcal{P}$-space, depending on the update type.

An ideal object such as the camel bust shown in Fig. 8.11 is placed in front of the TCP-mounted 3D-Modeller. In order to represent flawed surfaces, a glass screen is placed in between the object and the sensor in a second measurement. This setting is depicted in Fig. 8.12. The results are compared. While the exploration performance in simulations is already assessed in Chap. 6, the four update types are evaluated based on their suitability for real experiments in this section.

Figure 8.11: Camel bust and 3D-Modeller in order to evaluate simulation results with real objects.

Figure 8.12: Camel bust behind a transparent object.

In the following, results acquired by the SCS with the SGM stereo method are presented. The measurements are integrated into the $\mathcal{P}$-space represented as sphere-tree (cf. Sec. 5.5.3). The resolution of the $\mathcal{P}$-space is $12mm$. In these experiments, only a small slice of $\mathcal{P}_{unk}$ is examined. Measurements outside of the slice are omitted.



(a) Naïve Update (b) Belief Update

(c) Fuzzy Update (d) Bayes' Update

Figure 8.13: $\mathcal{P}$-space slice based on Naïve, Belief, Fuzzy, and Bayes' Update in sensing direction of the camel bust with (right) and without (left) measuring through a transparent object (glass screen).

The results for updating $\mathcal{P}$-space based on the four update types are depicted in Fig. 8.13. Object and background are an ideal surface for the SCS, resulting in a good approximation of the camel and the ideal surface in the background. The glass screen results in spurious measurement of points.

Discussion The update rules are validated. In Chap. 6, Bayes' and Fuzzy Update are selected due to their performance. However, the simulated results do not account for applicability in real sensings.

In these experiments, the sensor with lowest accuracy is used for assessment of the proper update rule. Belief and Naïve Update are not recommended for exploration. Bayes' and Fuzzy both recognize the glass object as an obstacle, which is the main requirement for the exploration task. Similar results are obtained for other non-ideal surface such as aluminium foil and green tissues. The foil is well detected by the SCS, the green tissue provides difficulties in detection for the LRS and LSP. Consequently, a combination of all three sensing principles using Bayes' or Fuzzy Update is recommended for exploration of realistic work cells aiming at robust determination of *safe-for motion* areas.

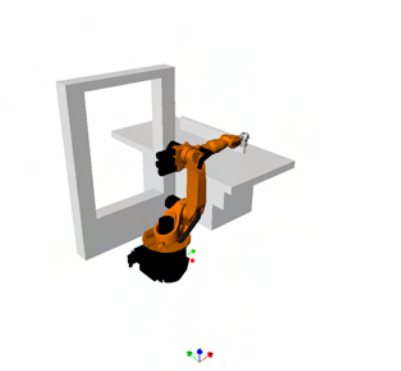


Figure 8.14: CAD-modeled SME scene.

Figure 8.15: SME scene with real objects.

8.3 Exploration of Configuration Space

In this section, experimental results from the exploration of a typical work cell are presented. In order to compare the experimental results to the simulations, the work cell objects are modeled in CAD and placed at the real locations in the $\mathcal{P}$-space representation.

The mission is to explore the $\mathcal{C}$-space of the robot, i.e. $w_{exp} = 1$, $w_{goal} = 0$, and $w_{RoI} = 0$. The goal is approached applying the different sensor combinations summarized in Tab. 8.2. The ideal sensor combination for a reliable assignment of $\mathcal{P}_{free}$ is elaborated in Chap. 6; consequently, the LRS is the main exploration sensor, supplemented with LSP and SCS.

Fig. 8.14 shows the CAD-modeled work cell. The work bench (cf. Fig. 8.1(c)) and frame (cf. Fig. 8.1(a)) are arranged in the work cell. The $\mathcal{P}$-space size of the corresponding directions is $\mathcal{P}(x, y, z) = [3000mm, 3000mm, 2000mm]$, the minimum sphere-tree resolution is $p_{res} = 50mm$.

The selected exploration strategy is PBS with AOD, $\lambda \in]0, \infty[$. Initially, $\mathcal{P}_{known} = 32\%$. The arrangement of $\mathcal{P}$-space is depicted in Fig. 8.16. The corresponding test-bed work cell is presented in Fig. 8.17. The frame is not completely covered by $\mathcal{P}_{unk}$ in order to demonstrate that initially known free areas can be changed into occupied regions, as the frame is not included in Fig. 8.17.

The results are exemplarily shown for the following sensor set-up: view planning is performed using the LRS, sensing is done by LRS and SCS, applying the SGM stereo method. In the real experiments, a median filter of window size 7x7 is applied in order to filter the stereo range data.

Tab. 8.2 documents that a majority of the time required for exploration is consumed by measurement including travel time. The robot's speed is limited to 10% of its maximum velocity of $4\frac{m}{s}$ in the experiments.

Figure 8.16: SME cell before simulated exploration.

Figure 8.17: SME scene before experimental exploration

Table 8.2: Exploration experiments. The joint limit set [T] in Tab. 8.1 is used. The work cell is depicted in Fig. 8.16. The experiments and simulations have been performed on a Linux PC with Dual Xeon Processor with 3.06 GHz and 1GB RAM. The table shows the mean of the time elapsed per iteration , the sensing time percentage is relative to the total time required per iteration.

Scene	Iterations	Planning Time [s]		Update Time [s]		Sensing Time [%]
		sim.	exp.	sim.	exp.	exp.
Fig. 8.16	10 15	324 400	206	323 271	181	72
Fig. 8.34: C1,C2	27	125	98	133	196	-
Fig. 8.31: C8,C9	12	195	71	155	166	-

The time required for re-sampling of grey nodes and updates of the roadmap is summarized as update time. The planning time combines the effort required for building the IG representation and computing the NBV. An update of the roadmap is performed when no view node can be determined using PBS.

Fig. 8.18 shows the simulated work cell status after 5 iterations. The corresponding result from the real experiment is presented in Fig. 8.19. The frame has been detected by the system and the region a positive occupancy state is assigned.

The results for a complete simulated exploration are presented in Fig. 8.20. The $\mathcal{C}$-space exploration process is depicted in Fig. 8.21.

Discussion The differences between the simulations and experiments can be explained as follows: In the simulations, measurements are considered noisy, but are assumed to detect obstacles, i.e. $p_{true} = 1$. In the real measurements, not all obstacles are detected by the LSP and SCS. This case is presented in Fig. 8.22. The polystyrene work bench is sensed with LSP and SCS. The surface is white, yet structured. As the LSP's laser-line is not fully segmented, less information is obtained. The SCS has the same drawback. The images acquired by the SCS are presented in Fig. 8.22(b). The resulting disparity image

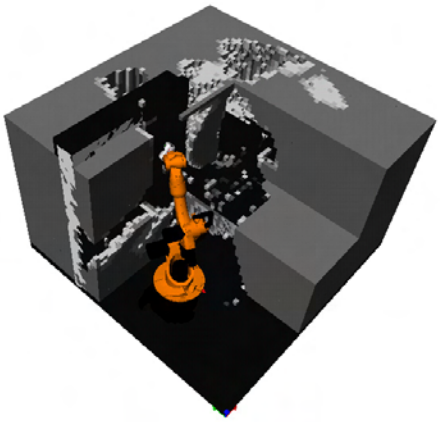


Figure 8.18: SME cell after 5 simulated exploration iterations with PBS, sensing with LRS and SCS, planning with LRS.

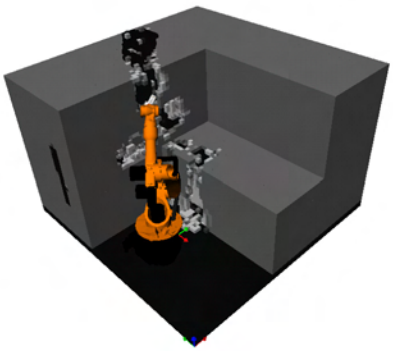


Figure 8.19: SME cell after 5 experimental exploration iterations with PBS, sensing with LRS and SCS, planning with LRS.

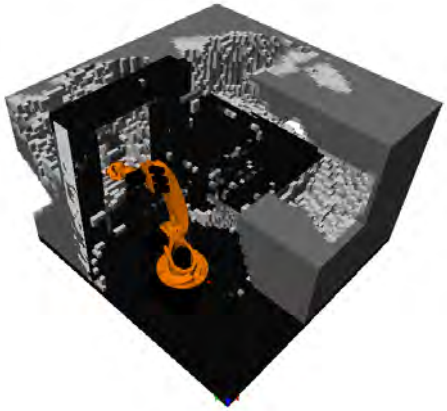


Figure 8.20: SME cell after 15 simulated exploration iterations with PBS, sensing with LRS and SCS, planning with LRS.

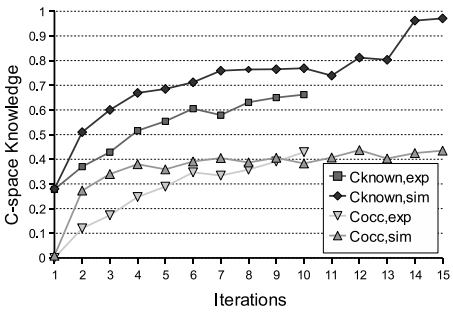
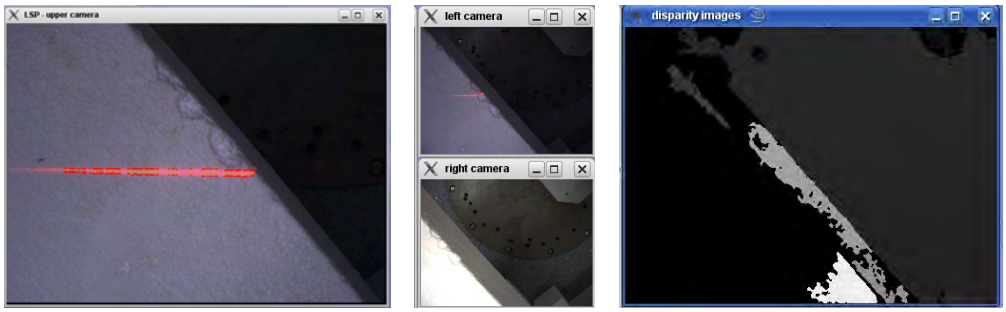


Figure 8.21: $\mathcal{C}$-space exploration results for SCS and LRS. Simulation and experiment are compared.

in Fig. 8.22(c) shows that only part of the cell is measured. Therefore, the IG per iteration is lower in the real exploration than in the simulation. However, *safe-for-motion* areas are reliably detected, as SCS and LSP do not change the state of the $\mathcal{P}$-space cells if no distance measurement is obtained.

Noise in the sensings, especially for the SCS, is reduced by a median filter. However, false detection of obstacles occurs. The upper right corner of the frame object in Fig. 8.19 shows a tail of the corner. This falsely detected obstacle is caused as follows: the room's lighting (cf. Fig. 8.15) is directly located above this corner, resulting in overexposure of the cameras, leading to erroneous disparity maps. This region is consequently falsely classified as being occupied, which limits the moveability of the robot. However, collisions with work cell objects do



(a) Segmented laser-line (b) Camera images (c) Disparity image

Figure 8.22: Measurement of the real work bench object, performed with LRS and LSP. The laser-line cannot be segmented in the complete image, therefore the maximal amount of depth information cannot be obtained. To the right, the disparity image of the SCS using SGM is presented. It demonstrates that large areas in the image do not provide depth information.

not occur. Further, the region is again reversed into $\mathcal{P}_{free}$ when sensed free from a different pose.

Further experiments include the combinations in Tab. 8.2. In the next section, an experimental exploration of a RoI containing a small bust, is presented.

8.4 Exploration of Regions of Interest

In this section, results from the exploration of a RoI in $\mathcal{P}$-space are given. A scene consisting of the small Zeus bust depicted in Fig. 8.23 is explored. In order to guide the exploration process, a RoI enclosing the bust is manually defined. Simulations for the same scenario have been performed. They are presented in the appendix. The [T] set of joint limits is applied, limiting the robot to moving to its left from the home configuration. Fig. 8.23 shows a photograph of the test-bed. The $\mathcal{P}$-space resolution is 25 mm.

In order to compare the manual and automatic exploration, a voxel model is acquired manually, depicted in Fig. 8.25. The robot is moved to three different view points around the bust, the measurements are acquired using the SCS. The voxel model of the Zeus bust is obtained by accumulating measurements using the SCS into the $\mathcal{P}$-space, which is initially entirely known to be free. Thus, solely voxels containing measurements are accumulated using Bayes' Rule.

In order to prepare the exploration of the bust, the RoI must be defined. Fig. 8.26 shows the RoI w.r.t. to the robot. The work cell initially known to the robot is presented in Fig. 8.24.

Figure 8.23: Photograph of Zeus bust, arranged on the podium depicted in Fig. 8.1(b).

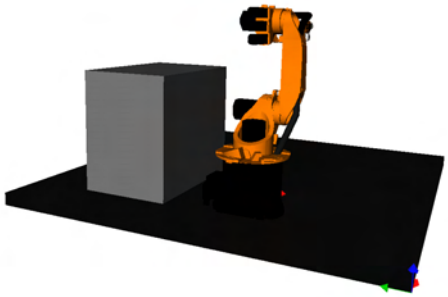


Figure 8.24: Representation of the Zeus scene in the planning environment. The unknown part of $\mathcal{P}$-space hides the Zeus bust and podium.

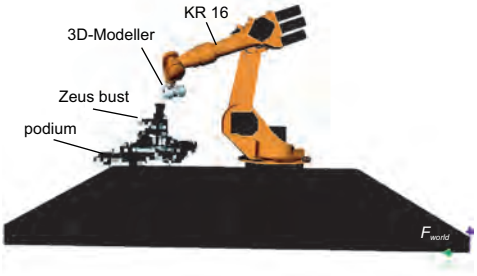


Figure 8.25: Zeus bust measured manually from three poses. The $\mathcal{P}$-resolution is 25mm

.

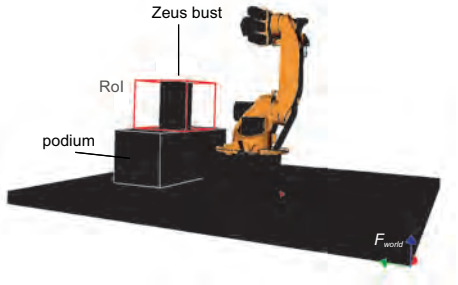
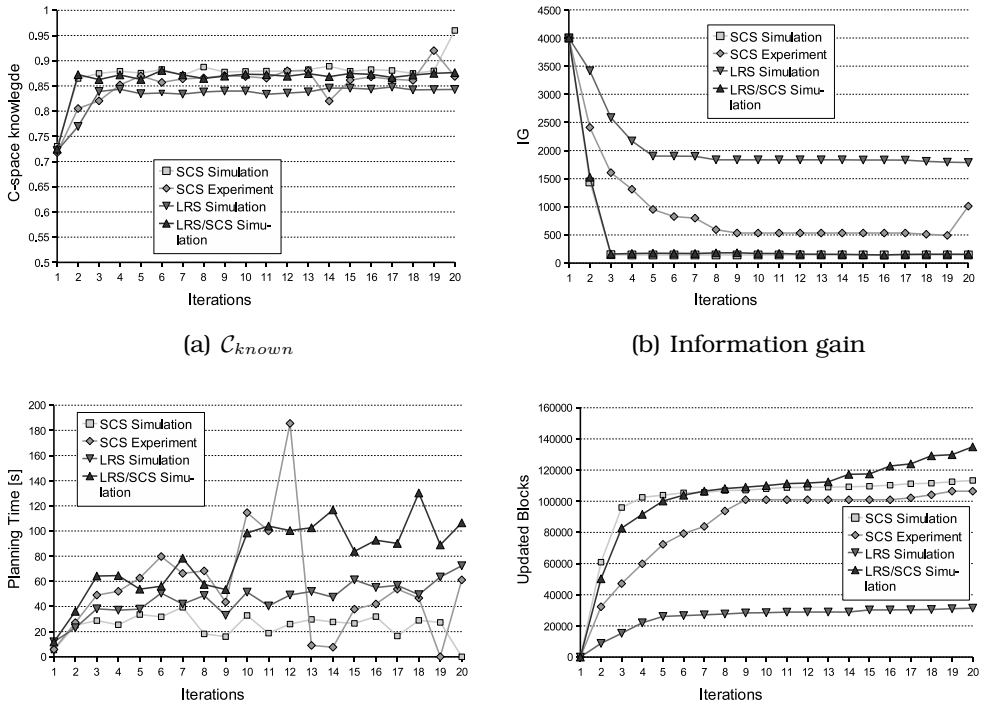


Figure 8.26: Zeus scene in the planning environment. The RoI and the bust are modeled. In the real experiment, merely the RoI is given. The Zeus and the podium are modeled as ideal surfaces for simulation purposes.

Figure 8.27: Photograph of robot and Zeus bust in iteration 17.

Figure 8.28: Representation of the Zeus scene in the planning environment after 17 iterations.

Simulations with SCS, LRS, and combined sensing with LRS/SCS are performed. Whenever the LRS is involved, it is used for planning views. The results are depicted in Fig. 8.29(a) to Fig. 8.29(d).



(a) $\mathcal{C}_{known}$ (b) Information gain

(c) Planning time (d) $\mathcal{P}$-space updates

Figure 8.29: Object exploration results for the Zeus scene depicted in Fig. 8.24 using Bayes' Update. The $\mathcal{P}$-space resolution is 25 mm, the RoI is depicted in Fig. 8.26. Simulations with SCS, LRS, and SCS/LRS are compared. The LRS is used for planning when applied for the task. An experiment with the SCS (SGM-method) is presented.

Discussion The development of $\mathcal{C}$-space knowledge is shown in Fig. 8.29(a). The mission is to explore a RoI, which results in a low information gain considering $\mathcal{C}$-space. The IG is computed based on the $\mathcal{P}$-space state inside the RoI, Fig. 8.29(b) shows the IG per iteration. The lower the remaining possible IG is, the more information has been acquired. The RoI is encompassed by $\mathcal{P}_{unk}$.

The Point planning part of PBS cannot place a configuration with FoV midpoint accordingly (it requires IG blocks on the boundary of $\mathcal{P}_{free}$). This results in application of the beam method in the initial exploration steps. The LRS has merely 30 percent of the measurement range of the SCS, thus the SCS plans most efficiently with PBS in the inspection scenario in this section. due to the small size of $\mathcal{P}_{unk}$, the SCS obtains a range measurement at all times, thus the planned views result in IG.

The LRS/SCS sensing combination plans the views based on the LRS, resulting in less efficient exploration due to the short sensing range. The real experiment performed with SCS shows a difference between simulated and experimental exploration. This difference can be explained by missed sensor readings of the SCS, which significantly reduce the exploration performance.

Table 8.3: Experiments in the partially known SME cell. The $\mathcal{P}$-space resolution is $r_P = 50mm$, the IG and CD level are $l_{CD} = l_{IG} = 0$. View planning is performed with LRS, sensings are acquired using LSP and LRS.

Experiment	Update	w_{RoI}	w_{exp}	NBV	real./sim.	Scene
C1	Bayes	0.8	0.2	PBS	real.	Fig. 8.34
C2	Bayes	0.8	0.2	PBS	sim.	
C3	Fuzzy	0.8	0.2	PBS	sim.	
C4	Naïve	0.8	0.2	PBS	sim.	
C5	Bayes	0.8	0.2	BRS	sim.	
C6	Bayes	0.0	1.0	BRS	sim.	
C7	Bayes	0.0	1.0	PBS	sim.	
C8	Bayes	0.0	1.0	PBS	real.	Fig. 8.31
C9	Bayes	0.0	1.0	PBS	sim.	

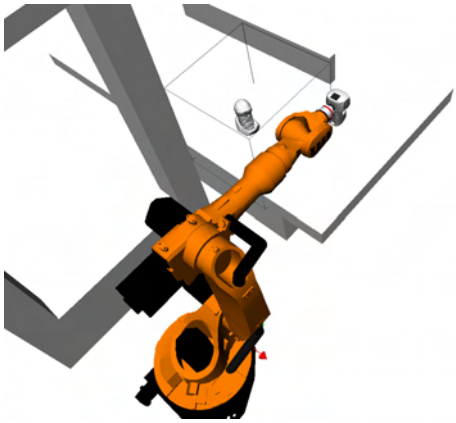


Figure 8.30: RoI and work cell of the combined exploration task.

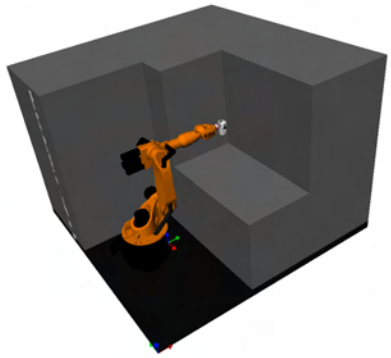


Figure 8.31: Initial $\mathcal{P}$-space in the combined $\mathcal{C}$-space and RoI exploration task.

8.5 Multiple Task Exploration

In this section, an experiment for a combined $\mathcal{C}$-space and $\mathcal{P}$-space exploration task is presented. The 3D-Modeller's LRS and LSP are used to explore a RoI enclosing the Zeus bust. Furthermore, the frame and the work bench are arranged in the work cell. The mission pursued in this experiment is to perform a weighted exploration of the $\mathcal{C}$-space and the RoI. The main focus is set on RoI exploration, yet information on the $\mathcal{C}$-space is acquired, i.e. $w_{exp} = 0.2$, $w_{goal} = 0$, and $w_{RoI} = 0.8$. The environment initially known to the robot is very limited (cf.Fig. 8.31). The experiments are summarized in Tab. 8.3, C1 – C4 are discussed in this section. The results from experiments C5–C9, confirming the findings from C1 and C2, are presented in the appendix (cf. Chap. B.7).

The LSP and LRS are used for performing measurements, the LRS is used for planning views by application of the PBS strategy. Simulative and realistic experiments are conducted.

In the experiment, only small parts of the work cell are known in advance. This is motivated by realistic applications in which the cell's objects are roughly known, yet their exact location is uncertain. Fur-

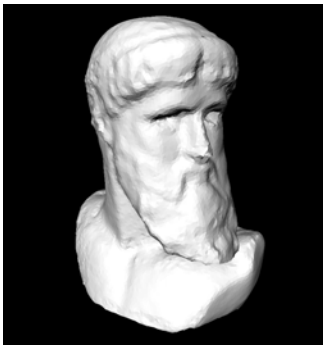


Figure 8.32: A 3-D surface model of the Zeus bust, acquired in manual mode using the LSP.

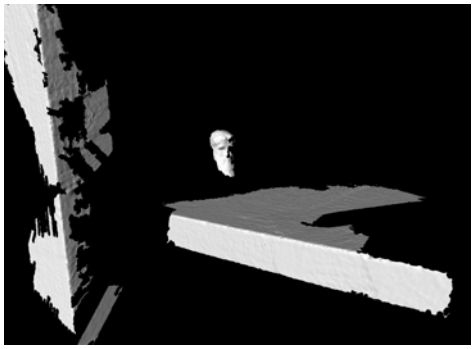


Figure 8.33: RoI including the Zeus bust, automatically modeled surface-based with the LSP.

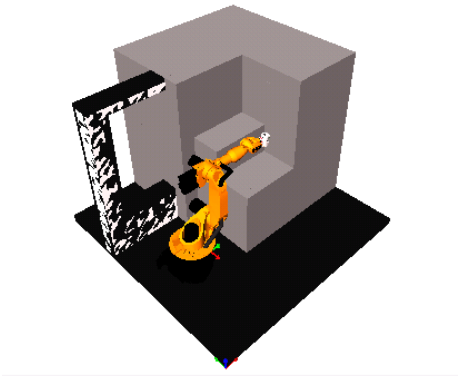


Figure 8.34: Partially known cell for simulated multiple task exploration.

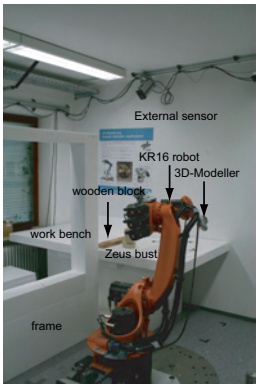
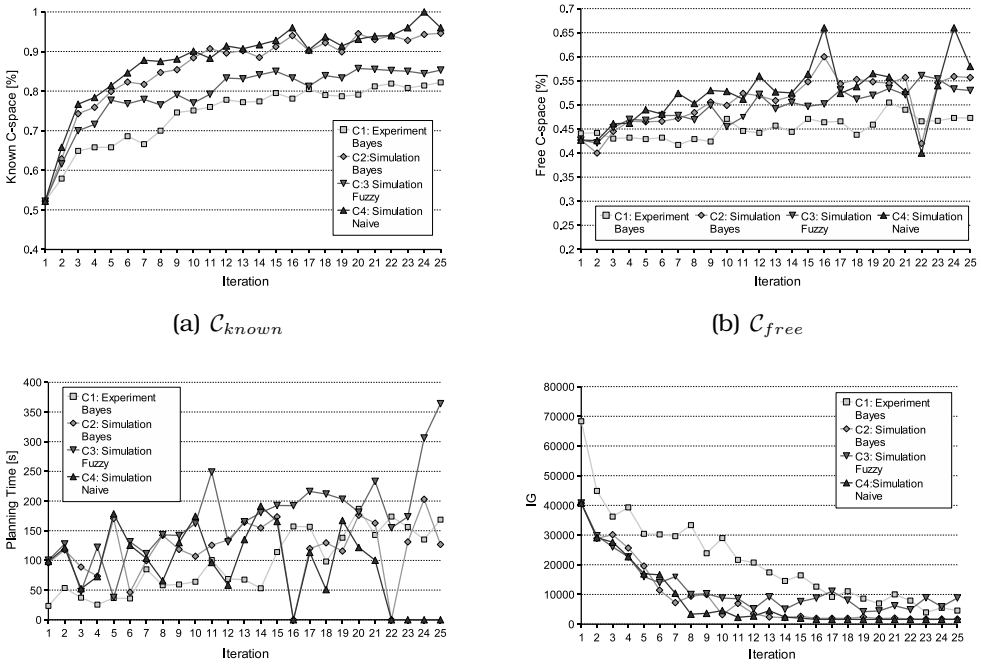


Figure 8.35: Test-bed setup for experimental multiple task exploration. Frame, work bench and Zeus bust are arranged in the test-bed. Additionally, the wooden block (cf. Fig. 5.3(a)) is inserted. The external cameras are not used in this experiment.

thermore, initially unknown objects are located on a work bench which must also be explored.

The location of the frame object in Fig. 8.1(a) is partially known, the exact location of the work bench is unknown. A RoI for the Zeus bust is defined on top of the work bench. As only an approximate arrangement of work bench and bust is known, the $\mathcal{P}$-space is initially specified to be unknown. In order to provide comparative results, the location of work bench and Zeus relatively to the robot and frame object is identical in the simulated and realistic experiment. In the simulations, work bench, frame, and Zeus bust are inserted as surface models. while in the experiments merely the frame is inserted.

The entire work cell is depicted in Fig. 8.34. The work bench and bust are hidden inside the unknown part of the cell. The frame is partially known and thus visible.



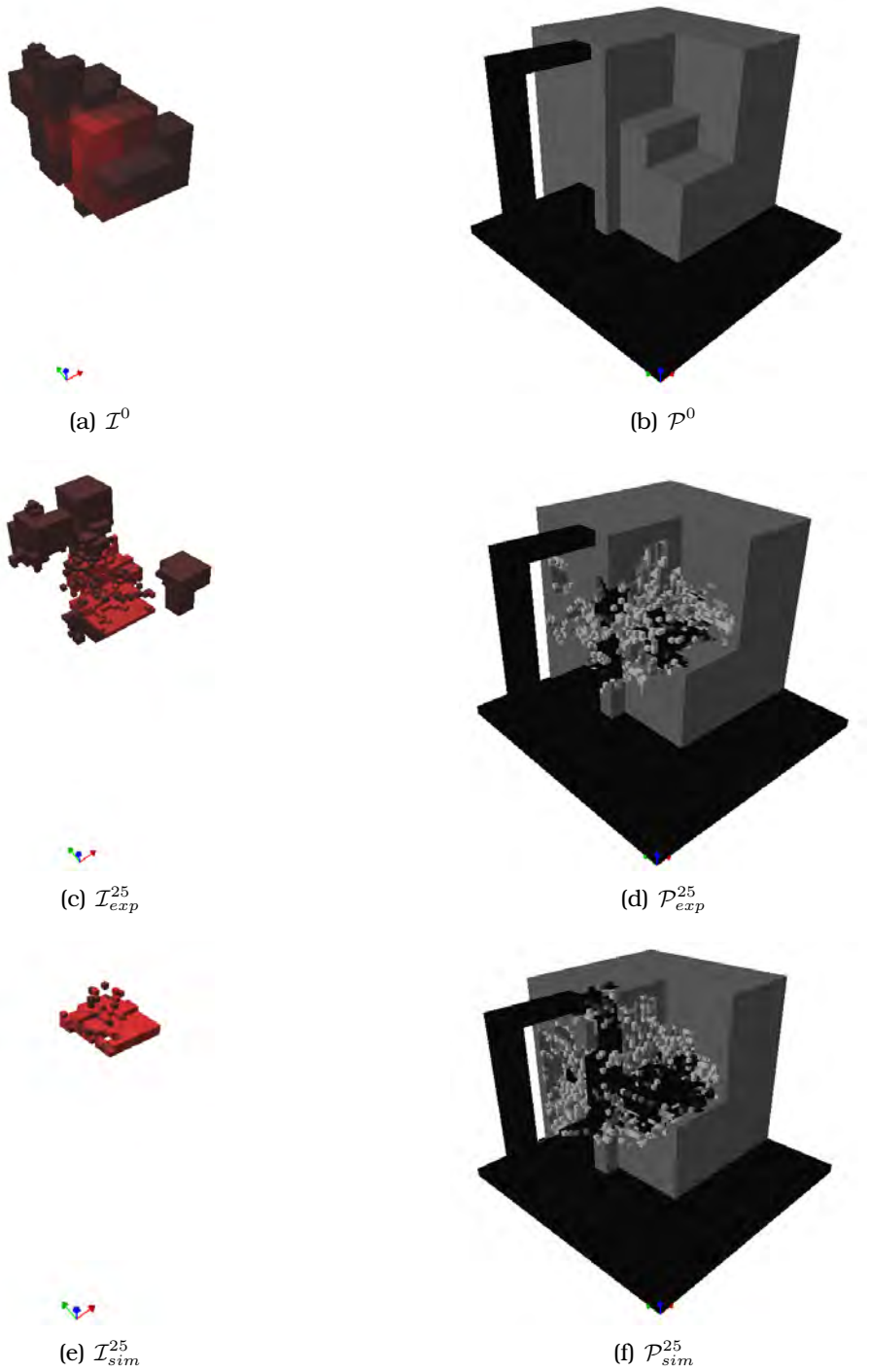
(a) $\mathcal{C}_{known}$ (b) $\mathcal{C}_{free}$

(c) Planning time (d) Information gain

Figure 8.36: Simulated and realistic experimental exploration results for the work cell depicted in Fig. 8.34. The mission is RoI and $\mathcal{C}$-space exploration, i.e. $w_{RoI} = 0.8$ and $w_{exp} = 0.2$, using the LRS for planning, supplemented with the LSP for sensings. Simulations with Fuzzy, Naive, and Bayes Update are presented, Bayes' Update is applied for the experiment.

The results of the experiments and their comparison to the simulations are presented in Fig. 8.5. In this figure, both $\mathcal{P}$-space and $\mathcal{I}$-map are presented.

Discussion As already identified in the previous experiments, the IG in the experiment is lower than in the simulation. However, the tendency is similar. The real exploration result achieves more than 80 % $\mathcal{C}$-space knowledge, the RoI is explored except for non-accessible, i.e. occluded, regions behind the bust, where the robot is unable to perform a measurement due to its kinematic constraints. This experiment demonstrates that partial knowledge of the work cell can be seamlessly combined with unknown parts. However, the Zeus bust, which is modeled surface-based (cf. Fig. 8.33) in parallel to the exploration, is not captured as good as the manually-scanned one in Fig. 8.32. Part of this effect can be explained by the fact that the planning was done with the LRS, while the surface model was acquired with the LSP.



(a) $\mathcal{I}^0$ (b) $\mathcal{P}^0$

(c) $\mathcal{I}^{25}_{exp}$ (d) $\mathcal{P}^{25}_{exp}$

(e) $\mathcal{I}^{25}_{sim}$ (f) $\mathcal{P}^{25}_{sim}$

Figure 8.37: IG and physical space for the work cell depicted in Fig. 8.34. The simulation (sim) and experiment (exp) are compared. The lighter the color of the IG blocks, the higher the IG.

8.6 Summary

In this chapter, experiments with a real robot system as well as further simulations are performed. The experiments address three major issues: the behavior of different update rules applied in real sensings, the optimal sensor combination, and the influence of the planning weights on the mission success.

Experiments with the update rule in combination with flawed surfaces show promising results. The Bayes' Update considers specular reflections, thus even the glassy object is detected as obstacle. The other update methods measure sparse points. The minimum goal, a *safe-for-motion* $\mathcal{P}$-space, can be guaranteed.

The simulations deliver better results than the real experiments. This is due to the fact that no measurements are rejected in the simulations. This does happen in the real-life case, resulting in lower performance. However, detailed experiments and assumptions on the environment properties and the corresponding measurements must be conducted in order to provide a realistic rejection and surface property model for each sensor. This effort exceeds the scope of this thesis.

The LSP supports the SCS in obtaining measurements of surfaces with little texture. The SCS provides measurements of glass surfaces, thus flawed surfaces can also be measured. However, the sensings of the SCS are prone to a higher level of errors than the other two sensors, resulting in measurements of obstacles at false locations. Therefore, the measurements from the SCS are median-filtered before they are accumulated into $\mathcal{P}$-space. Although a long-range sensor is beneficial for exploration when obstacles are detected, it must be supplemented with a distinct *safe-for-motion* sensor. Views are planned using the LRS as the outcome of its sensing corresponds best to the expected results. However, the measurement distance of the LRS is limited. Therefore, the LRS is joined with at least one of the longer-range sensors, either LSP or SCS, depending on cell size and resolution.

The automatic work cell exploration with PBS is efficient for the $\mathcal{C}$-space task. Objects can be modeled in a coarse voxel-based representation. When only a small scene must be explored, the SCS provides good results. Water-tight surface-based models of high resolution can not be obtained with this method automatically. In order to obtain surface models, an IG computation based on feedback from the surface modeling method will improve the performance.

9 Conclusion and Perspective

In this chapter, conclusive remarks on the achievements of this thesis and an outlook to applications and future research are provided. Current applications of the developed strategies and sensor design in the fields of medical robotics and cultural heritage preservation are described. Furthermore, an extension of the algorithm to exploration processes performed by mobile robots, i.e. robots with complex geometries on a mobile base (cf. Fig. 3.1), are elaborated.

9.1 Conclusion

In this thesis, a sensor-based exploration algorithm aiming for a maximum information gain is presented. Rather than mobile robots, complex kinematic stationary robots are used. The $\mathcal{C}$-space, i.e. maneuverability, of the robot system is optimally explored. The robot system is simulated; additionally, realistic experiments are performed. The exploration process is covered in its entirety, encompassing a theoretic assessment of the view planning problem, an extension to multiple missions, and a generalized architecture for autonomous exploration. The thesis presents a multiple task exploration using uncertainty in the planning stage.

In order to determine crucial exploration parameters, a novel simulation environment is developed. A sensor system concept is provided, addressing applications required in service robotics and flexible work cells. Concludingly, experiments with the designed system are presented, performed in a realistically modeled industrial work cell inspired by flexible work environments in SMEs.

The following sections recapitulate the key results assessed and applied in the development of this approach.

Comparison of Update Rules The data structure used in the thesis is a spatial 3-D occupancy grid. The grid is accumulated by imprecise sensings performed during exploration by the robot, therefore four update rules are evaluated in this thesis: Bayes', Belief, Fuzzy, and Naïve Update.

Naïve Update represents an ideal tristate grid: unknown, free, and occupied. In its current implementation, Belief Update is outperformed by the other update methods.

The remaining update rules are compared w.r.t. their exploration performance and integration of real sensings. Bayes' Rule is optimal in terms of planning performance and acquisition of non-perfect surfaces. However, many measurements do not contribute to the update due to measurement independence. Fuzzy Update has a promising potential for future use, as it does not require independence of cells and measurements. Although being fasted in measurement integration and view planning (it determines $\mathcal{P}_{known}$ rapidly), the Naïve Update is not recommendable in real experiments. Although most motion planning approaches apply this method, it is applied for reference purposes only in this thesis.

Entropy-Based Information Gain In order to perform multiple tasks simultaneously, a common measure for driving the exploration based on IG is applied. This allows an exploration of objects while simultaneously increasing the moveability of the robot. While in motion planning the $\mathcal{P}$-space is commonly assumed to be ideal, the uncertainty in the environment is considered when computing optimal sensor poses. Two view planning methods are developed in this thesis: the noisy beam method (BN) is specifically designed for $\mathcal{C}$-space exploration based on beam planning models considering sensor uncertainty. The Adaptive Obstacle Density (AOD) method uses the grey scale values of the occupancy grid for estimation of the intensity of an inhomogeneous Poisson Point Model, as opposed to the previously used homogeneous process, for computing the MER criterion in $\mathcal{C}$-space exploration.

As the latter outperforms the BN method while considering the uncertainty of the environment in the planning stage, it is recommended for use in exploration with occupancy grids. Furthermore, it enables a combined exploration of $\mathcal{C}$-space and $\mathcal{P}$-space simultaneously. In combination with the novel planning method developed in this thesis, an autonomous exploration of multiple targets simultaneously, considering sensor impreciseness, is achieved.

Development of a Planning Suite A novel sensor-based planning suite, the *SBP-Simulator*, is developed in this thesis. Experiments with the different Update Types are performed. Bayes' and Fuzzy Update show the best results in terms of planning time and determination of *safe-for-motion* areas in the chosen implementation. The occupancy grid is implemented as a sphere-tree, an efficient multi-scale spatial representation. By decoupling the data structure from collision detection and planning process, the approach can be applied in large physical spaces or bounded robot work cells in high resolution, while allowing affordable planning times.

Measurements are always accumulated on the lowest level, assuring correct data accumulation. A priori knowledge of the work cell and its components can be included; either provided as surface models or manually assigned as axis-aligned bounding boxes. Multiple sensors are readily usable, the software design is modular in order to allow an exchange of robots, sensors, and work cells. Uncertainty is considered depending on the update type.

For a seamless transfer of simulation results to real experiments, the SBP-Simulator is equipped with generalized interfaces for range sensors and robots. The range interface is implemented event-based, currently multiple 2-D and 2.5-D sensors can be connected simultaneously.

Assessment of Design Rules The SBP-Simulator is used to assess the design rules for a multi-mission exploration sensor. Safe exploration requires reliable determination of free regions of the physical space. Additionally, large sensing ranges are desired. Furthermore, multiple missions must be accomplished, requiring a multisensory vision system. In exploration, sensor and planning method are closely coupled. This thesis provides an evaluation of this constraint, elaborating that the following properties are required: safe determination of free regions, large FoV for object recognition, high accuracy for object modeling, and multiple sensing principles with overlapping FoVs in order to account for flawed surfaces. These design rules inspired the development of a multi-purpose vision platform, the 3D-Modeller.

This thesis addresses autonomous exploration in robot work cells. A sensor which is able to perform multiple missions in hand-eye configuration is missing to date. The 3D-Modeller is introduced as a multi-purpose vision platform, addressing most requirements in service robotics. Synchronization of multiple sensors is solved, exemplarily three sensing principles are fused: The laser-range scanner (LRS) for exploration, the laser-stripe profiler (LSP) for 3-D modeling, and the stereo camera sensor (SCS) for coarse environment acquisition and object recognition. The sensor design enables an extension by further sensors in order to continuously respond to latest developments.

The combination of laser-range scanner and laser-stripe profiler proves to be optimal for exploring medium-sized work cells at a high resolution. The stereo system combined with the laser-range scanner is optimal for capturing larger work cells with a lower resolution. The SGM method outperforms correlation-based methods such as MWMF, which is applied to object recognition.

Experimental Evaluation The developed sensor system is evaluated in a realistic test-bed, designed in analogy to a typical SME work cell. Experiments for pure $\mathcal{C}$-space exploration, exploration of RoIs, and combined missions are performed. The results assessed for $\mathcal{C}$-space exploration using the PBS planning method with AOD are convincing. This method enables other missions, e.g. view planning for object recognition or grasp planning.

In the following section, open problems and future research directions are described.

9.2 Perspective

While this thesis focuses on autonomous $\mathcal{C}$-space and $\mathcal{P}$-space exploration, the generality of the approach enables an application to other fields. In the following, further research is recommended, an extension to complex kinematic mobile systems as well as fields of application reaching beyond classic robotics are presented.

Further Research While most experiments successfully apply Bayes' Rule, the influence of the Update Type needs further evaluation; especially Fuzzy Update shows potential for an efficient exploration.

In simulations, all measurements are accepted. In the real life experiments, weaker exploration results are achieved, as not every sensed point results in a distance measurement. The modeling of these errors narrows the gap between simulation and real experiment. However, the modeling of missed measurements is difficult and requires assumptions on the surfaces properties. Initial research on using a rejection function biased by the surface properties are ongoing, however, not yet completed.

In this thesis, the view planning is performed for discrete view points. A computation of optimal paths for IG maximization seems promising. The developed architecture is suitable for an integration of sensings during path execution. The synchronization concept and software concept of the simulation environment are prepared for this integration, minor extensions are still to be implemented.

The view planning for object inspection can be further evaluated. An extension of the noisy beam method might produce less model error. Furthermore, the computation of the IG for object modeling derived from the surface model requires further research. Currently, no feedback from the surface modeling methods is used for the guidance of the exploration. However, the model-deficiency of a surface model, e.g. holes, might contribute information in order to compute the IG, i.e. the NBV. As perfect models do not exist in reality, the termination criterion of the exploration task requires careful consideration.

Sampling techniques are well-suited for acquisition of possible view nodes and navigation in roadmaps in exploration. The roadmap construction method could be optimized. Further sampling techniques, e.g. IG-based sampling, might be beneficial in order to increase performance.

A further improvement is the direct application of the sphere-tree for collision-free path planning. This reduces the memory required. However, the robot is then represented as spheres as opposed to the accurate surface model used by SOLID.

Enhancements to the sensor design are planned: The LSP should provide more information on laser line width and color detection, which could be used as confidence measure. Additionally, the use of cameras supported by Inertial Measurement Units (IMU) in order to design a self-referencing system is beneficial. The self-reference of the sensor can additionally compensated the pose error of the stationary Justin, as the robot's flexible joints result in pose inaccuracies in torque mode.

Extension to complex kinematic mobile systems Furthermore, application of the sensor system and its exploration capabilities to mobile systems with complex kinematics are envisaged. A mobile platform for the DLR humanoid robot Justin is being developed, leading towards a mobile system with more than 46 DoFs. The 3D-Modeller is one of the main sensors in this scenario, supported by 2.5-D range sensors based on Photonic Mixing Detectors (PMD) or structured light technology (cf. Fig. 2.5). Major challenges in this scenario are the extensions of this work towards consideration of pose uncertainty and localization. The sphere-tree data structure is extendable and allows for memory-efficient data integration. Therefore, its suitability for grid-based SLAM algorithms must be elaborated.

The Justin-scenario allows for an evaluation of the usability of the IG approach for object recognition. Additionally, non-static environments with moveable objects and their representations in $\mathcal{P}$-space must be evaluated.

Current Applications Experimental evaluations in medical applications of the 3D-Modeller show promising results. The LRS operates robustly in the challenging lighting conditions of Operation Rooms (OR). The task performed is the registration of patients (cf. Sec. E.1 for further details).

Another promising field of application is the preservation of cultural heritage. Data of different scales, ranging from airborne to close-range triangulation data, is required. In order to perform an efficient exploration, the largest scale data, i.e. airborne data, is used to determine an initial map for planning views. An insight into this non-robotic field of application is provided in the appendix, Sec. E.2.

The field of automation in SMEs provides numerous possible applications of exploration and perception. Automated bin-picking is a common robotic problem, which has not yet been solved satisfactorily. The task is to recognize the location of a bin and to identify arbitrary parts in the bin which must then be grasped. When applying 2.5-D sensors for generalized bin-picking, an efficient exploration, i.e. view planning strategy, is required. The pose of the bin and the parts to be picked are initially unknown. This task is similar to the modeling task, in which a RoI is defined. The exploration architecture proposed in this thesis enables an integration of the object recognition task, which is required in order to solve the bin-picking challenge.

This thesis addresses the sensor-based exploration problem with complex kinematic robots to its full extend. Uncertainty of sensors is considered in the planning stage; an efficient 3-D spatial grid structure allows for decoupling measurement integration, view planning, and motion planning. An architecture for performing multiple missions, concurrently apparent in flexible work cells and service robotics, is presented. The novel SBP-Simulator integrates three selected missions, $\mathcal{C}$-space exploration, goal exploration, and exploration of RoIs. These missions are evaluated in simulated and real experiments using the multisensory 3D-Modeller, which is designed for the needs in flexible work cells, e.g. object recognition, surface modeling, and tracking. Real experiments are performed in a test-bed, designed according to an SME flexible work cell. The developed strategies and methods enable a robotic system to autonomously explore a work cell, driven by multiple mission targets. The theoretic findings and experiments enable an extension of the approach to further fields of application.

A

Mathematical Appendix

A.1 Basics in Statistics

Important notations and probabilistic facts used throughout this thesis are recapitulated in the following:

Random variables, i.e. X, can take multiple values according to probabilistic laws. They can be continuous or discrete, depending on the application. In this thesis, most random variables are defined continuously. The probability that X has a value x is denoted as

$$p(X = x) = p(x) \ . \tag{A.1}$$

Discrete probabilities sum up to 1, i.e.

$$\sum_x p(X = x) = 1, \ p(X = x) \geq 0 \ . \tag{A.2}$$

In continuous spaces, random variables can take continuous values; in most cases each random variable possesses a probability density function (PDF). In case of normal distribution, the PDF is a Gaussian

$$p(x) = (2\pi\sigma^2)^{-\frac{1}{2}} exp\{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\} = \mathcal{N}(x; \mu, \sigma^2) \tag{A.3}$$

with variance σ^2 and mean μ. If x is a vector, the normal distribution is multivariate

$$p(x) = \det(2\pi\Sigma)^{-\frac{1}{2}} exp\{-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\} \tag{A.4}$$

with Σ being the positive, semi-definite, and symmetric covariance matrix. In the discrete case

$$\sum p(x) = 1 \ . \tag{A.5}$$

In the continuous case the integral of the PDF is

$$\int p(x)dx = 1 \ . \tag{A.6}$$

A value of the PDF may be larger than 1 in this case, as opposed to discrete probability. In case of multiple random variables, the joint probability is denoted as

$$p(x,y) = p(X = x \text{ and } Y = y) \overset{\text{Ind.}}{=} p(x)p(y) \ , \tag{A.7}$$

where Ind. describes $p(x,y)$ in case X and Y are independent. Usually, random variables contain information about other variables, e.g. the state of a map given measurements. This conditional probability is denoted

$$p(x|y) = p(X = x|Y = y) = \frac{p(x,y)}{p(y)} \overset{Ind.}{=} p(x) \ . \tag{A.8}$$

The theorem of total probability states:

$$\begin{aligned} p(x) &= \sum_y p(x|y)p(y) \text{ (discrete), or} \\ p(x) &= \int p(x|y)p(y)dy \text{ (continuous).} \end{aligned} \tag{A.9}$$

If the independent random variables x and y are combined with another random variable z, this leads to the following equation:

$$p(x,y|z) = p(x|z)p(y|z). \tag{A.10}$$

Common features or statistics of random variables are mentioned in the following:

The expectation of a random variable is denoted

$$\begin{aligned} E[X] &= \sum_x xp(x) \text{ (discrete), or} \\ E[X] &= \int xp(x)dx \text{ (continuous).} \end{aligned} \tag{A.11}$$

Additionally, the covariance of X is defined as:

$$Cov[X] = E[X - E[X]]^2 = E[X^2] - E[X]^2 \ . \tag{A.12}$$

The concept of entropy, applied widely in this thesis, is introduced:

The entropy of a random variable with PDF $p(x)$ is given by the following expression:

$$H(x) = E[-\log_2 p(x)] \tag{A.13}$$

which is

$$\begin{aligned} H(x) &= -\sum_x p(x)\log_2 p(x) \text{ (discrete), or} \\ H(x) &= -\int p(x)\log_2 p(x)dx \text{ (continuous).} \end{aligned} \tag{A.14}$$

A.2 Bayes' Rule

Bayes' Rule plays a predominant role in probabilistic inference. It relates the conditional probability of $p(x|y)$ to its 'inverse' $p(y|x)$

$$\begin{aligned} p(x|y) &= \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\sum_{x'} p(y|x')p(x')} \text{ (discrete)} \\ p(x|y) &= \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\int p(y|x')p(x')dx'} \text{ (continuous) .} \end{aligned} \tag{A.15}$$

In robotics, the probability $p(y|x)$ is often called *generative model* as it describes how a state variable X conditions the measurement Z. The denominator, which does not depend on x, is called *Normalizer of Bayes' Rule*.

In robotics, this denominator is often neglected [47], as its computation is quite complex and exhaustive. When updating a map, the entire map would have to be considered, which is intractable. Neglecting the denominator leads to inconsistencies in the map, especially if the same object is recorded in two consecutive measurements.

Therefore, recent work of Konolige [84] recommends the use of Log-Odds, detailed in the following:

An event w with the probability $p := P(w)$ has the Odd

$$O(p) := \frac{p}{1-p} = \frac{P(w)}{P(\bar{w})} \ . \tag{A.16}$$

Further, the Log-Odd is defined as

$$l_w = \log(O(p)) \ . \tag{A.17}$$

The Odd $\frac{P(A|B)}{P(\bar{A}|B)}$ and the Likelihood Quotient LQ $\frac{P(A|B)}{P(A|\bar{B}}$ must be distinguished. The Likelihood Ratio is the logarithm of the LQ, i.e. $lq = \log LQ$.

Log-Odds allow for a correct computation of the probabilities, as they evade the computation of the denominator, i.e. the scale. The inconsistencies resulting from the assumption of a constant scale, identified as the major drawback for the application of Bayes' Update [164] based on the inverse sensor model $p(y|x)$ for map building, do not appear when using Log-Odds.

The probability p can be derived from the Log-Odd as follows:

$$p = 1 - \frac{1}{1 + exp(l_w)} \ . \tag{A.18}$$

A.3 Dempster-Shafer Theory

The Dempster-Shafer Theory is based on two notions: the idea of obtaining degrees of belief from subjective probabilities for a question,

and Dempster's Rule for combining such degrees of belief when they are based on independent items of evidence. The *Frame of Discernment* contains all the hypotheses, each having its own *mass function* denoting the corresponding belief.

Frame of Discernment The frame of discernment Θ is the set of all opposing hypotheses

$$\Theta = \{A, B, C\} \ . \tag{A.19}$$

The set is considered complete, e.g. in describing measurements of a sensor, when it covers all possible sensor outcomes. Therefore, 2^Θ possible statements

$$2^\Theta = \{\emptyset, \{A\}, \{B\}, \{C\}, \{A, B\}, \{B, C\}, \{A, C\}, \{A, B, C\}, \Theta\} \tag{A.20}$$

exist.

Mass function A function

$$m : 2^\Theta \to [0, 1] \tag{A.21}$$

is called a mass function, iff

$$\begin{aligned} m(\emptyset) &= 0 \\ \sum_{X \in 2^\Theta} m(X) &= 1 \ . \end{aligned} \tag{A.22}$$

The mass function defines the support of a statement. If $m(\Theta) = 1$ the sensor does not produce plausible measurements, resulting in complete uncertainty, i.e. every statement is possible. As opposed to Bayes' Theory, the complement of a statement $\{A\}$ is the set $\{B, C\}$.

Dempter's Rule of Combination Combining two independent mass functions m_1 and m_2 results in a new mass function, based on the following rule of combination

$$m_1(X) \otimes m_2(X) = \begin{cases} \frac{\sum_{A_i \cap B_j = X} m_1(A_i) \cdot m_2(B_j)}{1 - \sum_{A_i \cap B_j = \emptyset} m_1(A_i) \cdot m_2(B_j)} & \text{for } X \neq \emptyset, A_i \in 2^\Theta, B_j \in 2^\Theta \\ 0 & \text{for } X = \emptyset \end{cases} . \tag{A.23}$$

The denominator serves normalization purposes, where

$$k = \sum_{A_i \cap B_j = \emptyset} m_1(A_i) \cdot m_2(B_j) \tag{A.24}$$

denotes the inconsistency of the data sources. If $k \to 1$ the results are not interpretable, for $k = 1$ the rule is not defined. The numerator

$$l = \sum_{A_i \cap B_j = X} m_1(A_i) \cdot m_2(B_j) \tag{A.25}$$

denotes a measure for the hypothesis-supporting evidence. If there are more than two sources of information, they are combined subsequently. In reality, high contradiction of hypotheses with high reliability rarely occurs.

Belief Function Given a mass function m, a function

$$Bel : 2^{\Theta} \rightarrow [0,1] \tag{A.26}$$

is called *belief function*, iff

$$Bel(X) = \sum_{\substack{A \subseteq X \\ A \neq \emptyset}} m(A) \ . \tag{A.27}$$

This function measures the total support, i.e. belief, in X.

Plausibility Function Given a mass function m, a function

$$Pl : 2^{\Theta} \rightarrow [0,1] \tag{A.28}$$

is called *plausibility function*, iff

$$Pl(X) = \sum_{A \cap X \neq \emptyset} m(A) \ . \tag{A.29}$$

This function measures the total possible support, i.e. plausibility, in X. The interval $Bel - Pl$ represents the uncertainty: the larger the interval, the less trustworthy the statement. The interval $[0,1]$ corresponds to complete uncertainty, the smaller the interval, the safer the statement. Ideally, the interval degenerates into a point.

A.4 Fuzzy Set Theory

In this section, the fuzzy set theory is briefly described. Basic properties as well as combination rules used throughout this thesis are introduced. Further details can be obtained from [122].

In the classic set theory, a subset $A \subset G$ of a basic set G can be defined by an indication function

$$I_A : \left\{ \begin{array}{ccc} G & \rightarrow & \{0,1\} \\ g & \mapsto & I_A(g) \end{array} \right.$$

as

$$A := \{g \in G : I_A(g) = 1\} \ .$$

Intersection, union, and complements of sets can be defined via indications:

$$A \cap B = \{g \in G \mid I_A(g) = 1 \wedge I_B(g) = 1\} = \{g \in G \mid min\{I_A(g), I_B(g)\} = 1\}$$

$$A \cup B = \{g \in G \mid I_A(g) = 1 \vee I_B(g) = 1\} = \{g \in G \mid max\{I_A(g), I_B(g)\} = 1\}$$

$$\bar{A} = \{g \in G \mid 1 - I_A(g) = 1\}.$$

This set comprehension can be generalized by choosing membership functions, e.g. $\mu_A : G \to [0,1]$ rather than indication functions. The corresponding fuzzy set $\tilde{A}$ is the defined as

$$\tilde{A} := \{(g, \mu_A(g)) : g \in G\}.$$

If the related basic set G is known, then $\tilde{A}$ is identified by μ_A. In analogy to the classic set theory, intersection, union and complement are defined by

$$\begin{aligned} \forall g \in G: & \\ \mu_{\tilde{A} \tilde{\cap} \tilde{B}}(g) := & \quad min\{\mu_A(g), \mu_B(g)\} && \text{(A.30)} \\ \mu_{\tilde{A} \tilde{\cup} \tilde{B}}(g) := & \quad max\{\mu_A(g), \mu_B(g)\} && \text{(A.31)} \\ \mu_{\overline{\tilde{A}}}(g) := & \quad \{1 - \mu_A(g)\}\ . && \text{(A.32)} \end{aligned}$$

The membership function μ_A assigns the degree of membership $\mu_A(g)$ of element g to set $\tilde{A}$ for every $g \in G$.

The degree of membership can not be interpreted as a probability: Probabilities represent the ignorance on an event, whereas in Fuzzy Sets the degree of membership is exactly known.

However, in the field of robotics and engineering, an interpretation of the degree of membership as a probability is often performed. Although the theory is violated, results are obtainable.

In the classic set theory, indication functions are used for the definitions above, therefore it can be considered a special case of fuzzy set theory. Intersection and union of fuzzy sets are generalized by applying t-norms or t-conorms instead of minima and maxima in Eq. (A.30) respectively Eq. (A.31). Furthermore, the complement operation is generalized by applying a negation instead of $f : x \mapsto 1 - x$ in Eq. (A.32).

B

Experimental Appendix

In this chapter, information on the simulations and experiments is summarized. The 3-D models of the robot system are depicted, its calibration is detailed. Further, plots of the $\mathcal{P}$-spaces and IG maps obtained during the simulations and experiments are presented per update rule. Concludingly, further results of the multiple task exploration experiments (cf. Sec. 8.5) are provided.

B.1 3-D Models of the Robot System

In this section, the 3-D models of sensors and robot used for visualization in the SBP-Simulator are depicted in Fig. B.1. Theses models are additionally used for computing collision-free paths applying the SOLID [171]. The 3D-Modeller Halfstep B.1(a) variant is an preliminary version of the final 3D-Modeller B.1(b); it is currently used to evaluate further sensing principles.

(a) 3D-Modeller Halfstep (b) 3D-Modeller (c) KR 16

Figure B.1: 3-D models of robot and sensors used in the experiments and simulations. The surface models are applied for visualization and collision detection.

B.2 Calibration

In this section, the steps required for the calibration of the sensors on the robot are summarized. Additionally, further information on the test-bed setup of the real robot system is provided.

B.2.1 Hand-Eye Calibration

All sensors must be calibrated on the robot, i.e. hand-eye calibration. In order to transform the range sensings $\mathbf{d}_{S,i}$ of sensor i into the robot base coordinate system $\mathbf{d}_{B,i}$, the sensors must to be calibrated on the tool center point of the robot (TCP). The transformation ${}_T\mathbf{T}_i{}^S$ is specific to each sensor in the 3D-Modeller. The whole process is performed semi-automatically, however, complete automation is also possible.

$$\mathbf{d}_{B,i} = {}_B\mathbf{T}^T \cdot {}_T\mathbf{T}_i{}^S \cdot \mathbf{d}_{S,i} \tag{B.1}$$

In order to set up the system, three calibration steps are required:

1. intrinsic and extrinsic (hand-eye) calibration of the two cameras;
2. calibration of the laser plane(s) of the (dual) LSP laser projector;
3. calibration of the LRS-to-TCP transformation.

The camera calibration is performed with CalDe/Callab [150, 187]. A calibration programm automatically moves the robot to poses with varying inclination angles and distances to the planar checker board pattern. Corresponding images are taken in order to compute the camera's intrinsic and extrinsic parameters with sub-pixel accuracy.

The next step is the calibration of the laser-projectors of the LSP w.r.t. the cameras. As the cameras are calibrated beforehand, it suffices to sweep the LSP at at least two different distances and varying inclination angles over the same plane used for the camera calibration, acquiring images. Details on this process are described in [151]. An automatic robot program performs the acquisition of calibration data.

Finally, the transform of the LRS to TCP is estimated. The calibration is performed by either of two procedures:

- Sensings on a sphere with exactly known diameter are acquired, a least square method is applied to estimate the center of the sphere and the transform ${}_{TCP}\mathbf{T}_i{}^S$ is obtained. This method [154] is optimized for hand-guided operation. Results for the sphere calibration with the LRS are presented in Fig. B.2.
- Alternatively, the identical procedure as stated for the laser projector above is performed, as the teaching of a calibration trajectory around the sphere with distances of approximately 100mm to the sphere surface is challenging using the robot.

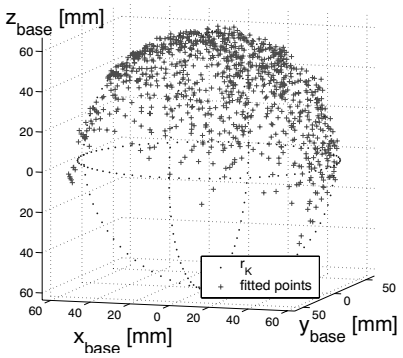


Figure B.2: Results from the LRS calibration on a sphere.

In order to reduce the calibration effort when mounting the system to a new pose sensor (robot) the individual ${}_{TCP}\mathbf{T}_i{}^S$ can be obtained by a method called TCP re-calibration. The relative pose between the sensors does not change when re-mounting the 3D-Modeller. Therefore, the new transforms can be obtained by simply performing step 1 of the calibration process, obtaining a new ${}_{T}\mathbf{T}_i{}^S$ for the cameras.

$$\underbrace{{}_{T,n}\mathbf{T}_i{}^S}_{\text{new step 1}} = {}_{T,n}\mathbf{T}^{T,o} \cdot \underbrace{{}_{T,o}\mathbf{T}_i{}^S}_{\text{old step 1}} \Rightarrow {}_{TCP,n}\mathbf{T}^{T,old} = {}_{T,n}\mathbf{T}_i{}^S \cdot {}_{T,o}\mathbf{T}_\mathbf{i}{}^{S^T} \quad \text{(B.2)}$$

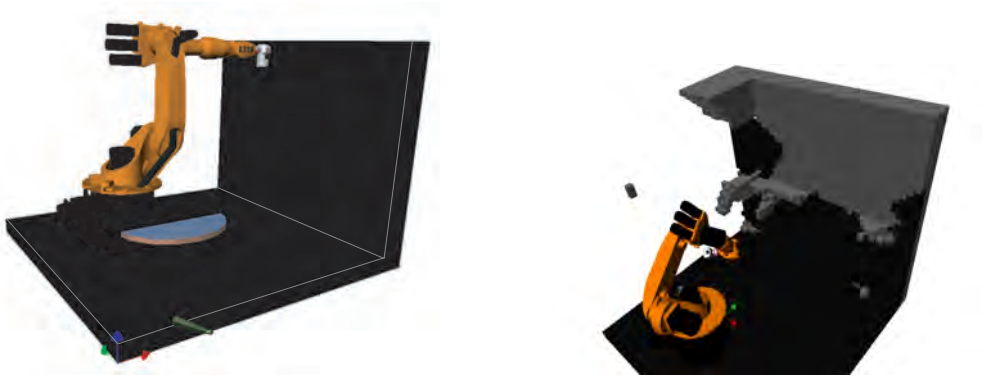
The transform ${}_{TCP,n}\mathbf{T}^{T,old}$ is used to re-calibrate the LSP and LRS. It is important to know the system the original calibration was performed on, as e.g. a calibration on a low-accuracy tracking systems and a later re-calibration on a highly accurate robot is not recommended.

B.2.2 Work Cell Coordinate Frame

In the planning tool, a world coordination system with index W is defined in the lower right corner of the work space. In order to set up the cell correctly, the transform ${}_{W}\mathbf{T}^B$ from robot base to world frame must be estimated. If the robot is not limited in its joint angles, the work cell must cover the entire work space of the robot. The area outside the modeled $\mathcal{P}$-space is assumed *safe-for-motion*. The maximum range of the robot in three dimensions is denoted $\mathbf{r}_{max} = [r_{x,max}, r_{y,max}, r_{z,max}]$. If the robot is placed in the middle of the work space, then ${}_{W}\mathbf{T}^B = [\ddots, \mathbf{r}_{max}]$, the rotational part depends on the application.

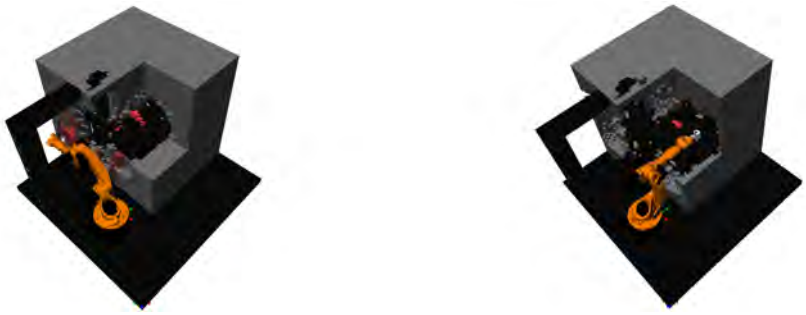
B.3 Naïve Update

In this section, figures of the $\mathcal{P}$-space obtained in the simulations and experiments applying Naïve Update are summarized.



(a) Original Scene (b) $\mathcal{P}^{5}_{Naïve}$

Figure B.3: Physical space of the work cell depicted in Fig. 6.17 on page 130 using Naïve Update.



(a) $\mathcal{P}^{10}_{Naïve}$ (b) $\mathcal{P}^{25}_{Naïve}$

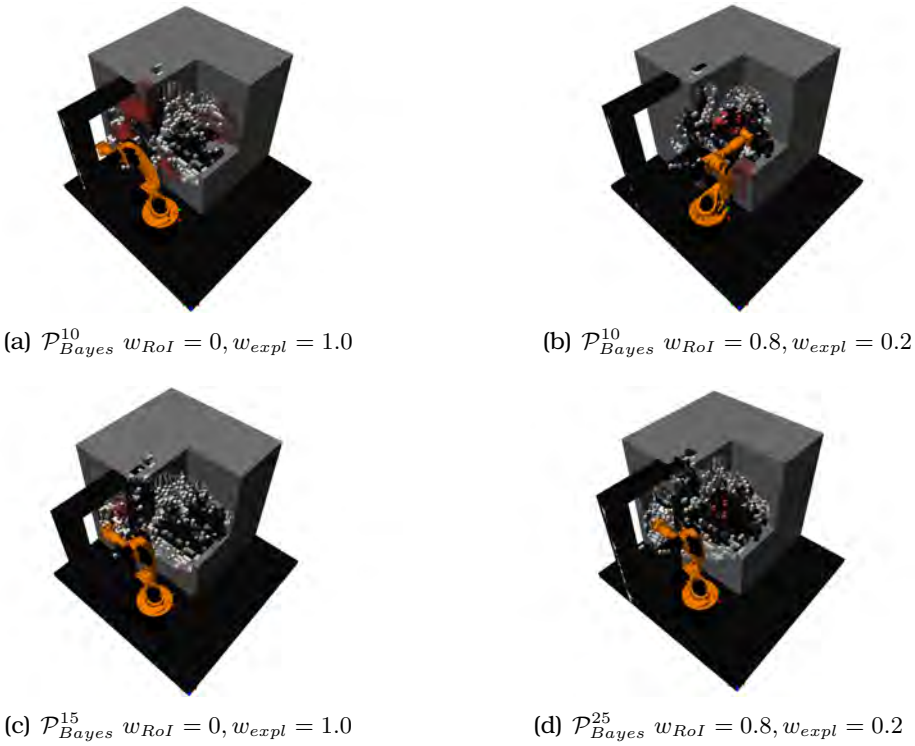
Figure B.4: Work cell with multiple exploration tasks using Naïve Update. Views are planned with LRS applying PBS, measurements are obtained using LRS and LSP. The exploration weights $w_{RoI} = 0.8$ and $w_{expl} = 0.2$ are assigned (cf. C4 Tab. B.1), the work cell is depicted in Fig. 8.34 on page 130. RoI, IG, robot, and $\mathcal{P}$-space are shown.

B.4 Bayes' Update

In this section, figures of the $\mathcal{P}$-space obtained in the simulations and experiments applying Bayes' Update are summarized. The cell's surface, which is first intersected by the sensing beam, is considered the surface location, as all cells are assumed to be filled completely.

(a) $\mathcal{P}^{5}_{Bayes}$ (b) $\mathcal{P}^{10}_{Bayes}$

Figure B.5: Physical space of the work cell depicted in Fig. 6.17 on page 130 using Bayes' Update.



(a) $\mathcal{P}^{10}_{Bayes}$ $w_{RoI} = 0, w_{expl} = 1.0$ (b) $\mathcal{P}^{10}_{Bayes}$ $w_{RoI} = 0.8, w_{expl} = 0.2$

(c) $\mathcal{P}^{15}_{Bayes}$ $w_{RoI} = 0, w_{expl} = 1.0$ (d) $\mathcal{P}^{25}_{Bayes}$ $w_{RoI} = 0.8, w_{expl} = 0.2$

Figure B.6: Work cell with multiple exploration tasks using Bayes' Update. Views are planned with LRS applying PBS, measurements are obtained using LRS and LSP. C2 and C7 are compared (cf. Tab. B.1), the work cell is depicted in Fig. 8.34 on page 177. RoI, IG, robot, and $\mathcal{P}$-space are shown.

(a) $\mathcal{P}^{10}_{Bayes}$ $w_{RoI} = 0, w_{expl} = 1.0$

(b) $\mathcal{P}^{10}_{Bayes}$ $w_{RoI} = 0.8, w_{expl} = 0.2$

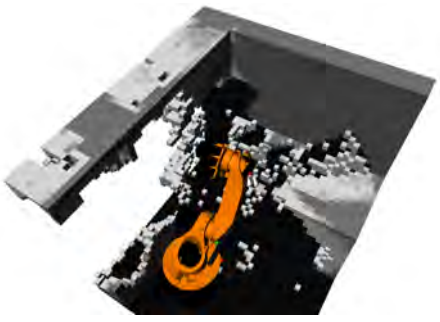
(c) $\mathcal{P}^{20}_{Bayes}$ $w_{RoI} = 0, w_{expl} = 1.0$

(d) $\mathcal{P}^{25}_{Bayes}$ $w_{RoI} = 0.8, w_{expl} = 0.2$

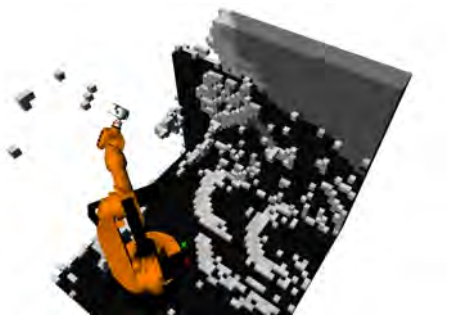
Figure B.7: Work cell with variable exploration tasks using Bayes' Update. Views are planned with LRS applying BRS, measurements are obtained using LRS and LSP. C5 and C6 are compared (cf. Tab. B.1), the work cell is depicted in Fig. 8.34 on page 177. RoI, IG, robot, and $\mathcal{P}$-space are shown.

B.5 Belief Update

In this section, figures of the $\mathcal{P}$-space obtained in the simulations and experiments applying Belief Update are summarized. The measured surface is located in the cell's center. As opposed to Bayes' Update, where the cell's surface, which is first intersected by the sensing beam, is considered the surface location.



(a) $\mathcal{P}^{5}_{Belief}$



(b) $\mathcal{P}^{10}_{Belief}$

Figure B.8: Physical space of the work cell depicted in Fig. 6.17 on page 130 using Belief Update.

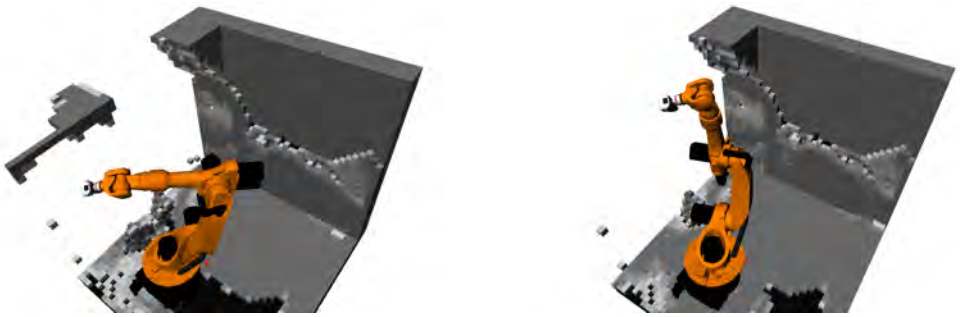
B.6 Fuzzy Update

In this section, figures of the $\mathcal{P}$-space obtained in the simulations and experiments applying Fuzzy Update are summarized.

A uniqueness of Fuzzy Update in its current implementation is its tendency to generate a layer of unknown cells on detected surfaces, in simulation as well as in real experiments. This effect can be explained by the computation of the cell's state $p(c_i)$ based on $m_{c_i}(free)$ and $m_{c_i}occ$:

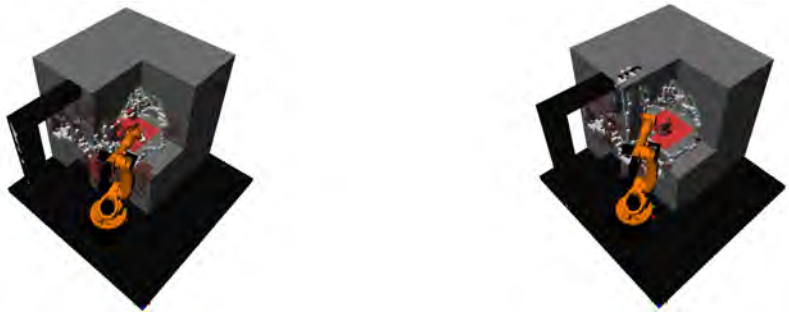
$$p(c_i) = \frac{m_{c_i}(occ) - m_{c_i}(free) + 1}{2} \ . \tag{B.3}$$

The measured surface is located in the cell's center. As opposed to Bayes' Update, where the cell's surface, which is first intersected by the sensing beam, is considered the surface location. This effect, although less obvious, is also present in Belief Update. The larger influence in Fuzzy is caused by the fact that $m_{c_i}(free)$ and $m_{c_i}occ$ are continuously added up.



(a) $\mathcal{P}^{5}_{Fuzzy}$ (b) $\mathcal{P}^{10}_{Fuzzy}$

Figure B.9: Physical space of the work cell depicted in Fig. 6.17 on page 130 using Fuzzy Update.



(a) $\mathcal{P}^{10}_{Fuzzy}$ (b) $\mathcal{P}^{25}_{Fuzzy}$

Figure B.10: Work cell with multiple exploration tasks using Fuzzy Update. Views are planned with LRS applying PBS, measurements are obtained using LRS and LSP. The exploration weights $w_{RoI} = 0.8$ and $w_{expl} = 0.2$ are assigned (cf. C3 Tab. B.1), the work cell is depicted in Fig. 8.34 on page 177. RoI, IG, robot, and $\mathcal{P}$-space are shown.

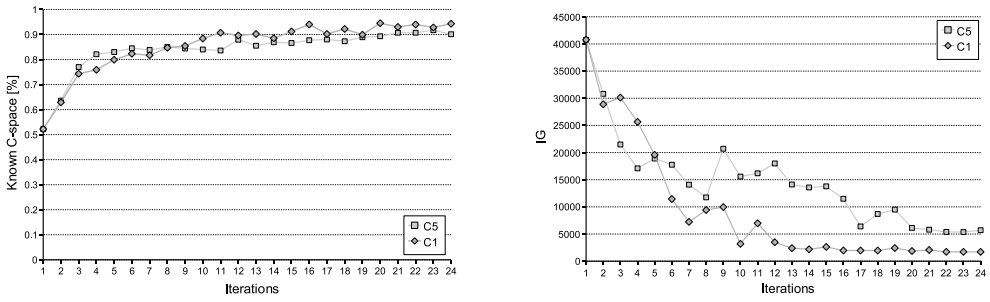
B.7 Multiple Task Exploration

In this section, the results of the experiments C3 – C9 (cf. Sec. 8.5) are presented. Experiments C3 – C7 are performed in the work cell with a partially known frame (cf. Fig. 8.34 on page 177), C8 and C9 are performed in a work cell in which the frame is completely enclosed by $\mathcal{P}_{unk}$ (cf. Fig. 8.31 on page 176). Tab. B.1 summarizes the parameters for the experiments.

Table B.1: Experiments in the partially known SME work cell. The $\mathcal{P}$-space resolution is $r_P = 50mm$, the IG and CD level are $l_{CD} = l_{IG} = 0$. View planning is performed with LRS, sensings are acquired using LSP and LRS.

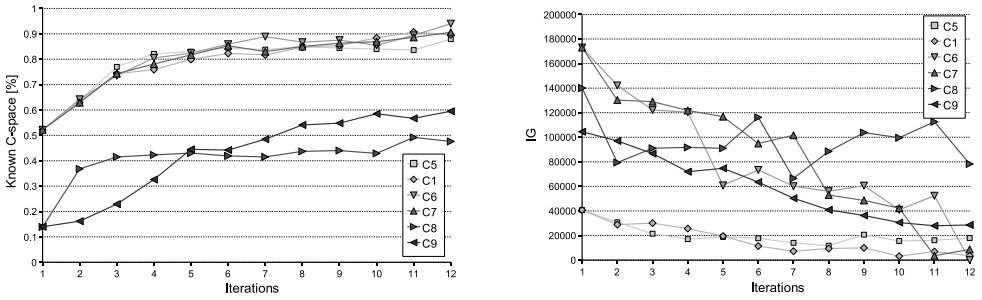
Experiment	Update	w_{RoI}	w_{exp}	NBV	real./sim.	Scene
C1	Bayes	0.8	0.2	PBS	real.	Fig. 8.34
C2	Bayes	0.8	0.2	PBS	sim.	
C3	Fuzzy	0.8	0.2	PBS	sim.	
C4	Naïve	0.8	0.2	PBS	sim.	
C5	Bayes	0.8	0.2	BRS	sim.	
C6	Bayes	0.0	1.0	BRS	sim.	
C7	Bayes	0.0	1.0	PBS	sim.	
C8	Bayes	0.0	1.0	PBS	real.	Fig. 8.31
C9	Bayes	0.0	1.0	PBS	sim.	

The mean planning times for the experiments in Tab. B.1 is depicted in Fig. B.7. The view planning time required by BRS is higher than that of the respective PBS method, as the BRS algorithm is more complex.



(a) $\mathcal{C}_{known}$ (b) IG

Figure B.11: Simulated exploration results for the work cell depicted in Fig. 8.34 on page 177. The mission is to explore the $\mathcal{C}$-space, i.e. $w_{RoI} = 0.0$ and $w_{exp} = 1.0$, using LRS for planning, supplement with LSP for sensings. Bayes' Update is applied for the simulations. The BRS and PBS planning methods are compared (cf. C6 and C7 in Tab. B.1).



(a) $\mathcal{C}_{known}$ (b) IG

Figure B.12: Simulated and experimental exploration results for the work cell depicted in Fig. 8.31 on page 176. Bayes' Update is applied for the simulations, BRS and PBS are used for view planning. (cf. Tab. B.1).

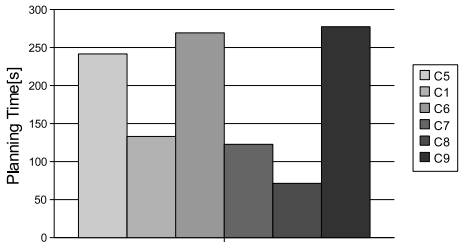


Figure B.13: Mean planning time for the explorations (cf. Tab. B.1).

C

Hand-Guided Multisensory 3D-Modeller

The ideas and experiences derived from early robotic application and the common lack of multisensory systems inspired the development of an hand-guided multisensory system.

It applies basic methods of synchronization, data fusion, and surface model generation. A synchronization concept for merging data acquired by various sensors is described. For pose measurement, a passive manipulator (FAROarm[1]) or an optical tracking system (ART-system[2]) is used.

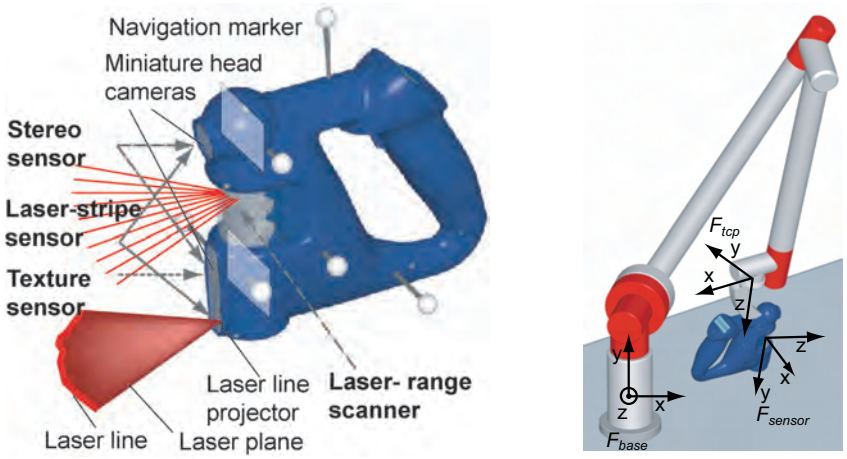
C.1 System Overview

Each 2-D sensor system needs to be spatially calibrated on the chosen pose sensor. Further, pose and sensor data need to be associated w.r.t. acquisition time. Due to the multiple kinds of sensors supported by the system, a reliable and extendable concept for time-synchronization is required. The 3D-Modeller integrates a laser-range scanner, a texture sensor (calibrated Charge-Coupled Device (CCD) miniature head camera), and a laser-stripe sensor, using the second camera in combination with a line laser module (opening angle 60°, 635 nm wavelength).

The laser-range scanner is based on the triangulation principle. It is specifically designed as a collision avoidance sensor for robotic appli-

[1] FARO Technologies http://www.faro.com LINK: 2007-08-15

[2] advanced real-time tracking (A.R.T.) GmbH, http://www.ar-tracking.de LINK: 2007-08-15



(a) Different sensor types (b) FAROarm

Figure C.1: Hand-guided 3D-Modeller for use with optical tracking system and passive manipulator.

cations. It causes no restrictions to the manipulation abilities of the robot it is attached to. Main features are the low weight, its robustness, and its large angle of view, which is variable from 0° to 270°. The measurement distance is 50 mm to 300 mm. It generates (externally triggered by a 25 Hz video synchronization pulse) the time stamp signals for the other sensors.

The different sensors are integrated into a specially developed housing. The housing is of low weight and easy to handle. Pose estimation is performed by the FAROarm or the ART-system. Here, the retroreflective markers as shown on the hand-guided device in Fig. C.1 are used for 6-D pose measurement. The pose sensors can be exchanged, as the system interfaces are unified in order to keep the system open to multiple sensors.

C.2 Sensor Synchronization

The DLR laser-range scanner communicates via Controller Area Network (CAN) bus, a priority-based field bus with real-time capabilities. Merging multiple sensors with multiple interfaces is a major challenge in sensor synchronization. This problem is solved by using the CAN bus as the master synchronization bus for time stamps. Each sensor is supplied with the same video synchronization pulse (generated e.g. by the video cameras), which results in synchronous measurements. The laser-range scanner generates its internal clock signal from this source. The scanhead rotates with 25 turns/second. The time stamp increases with each turn, so that every 40 ms a new time stamp mes-

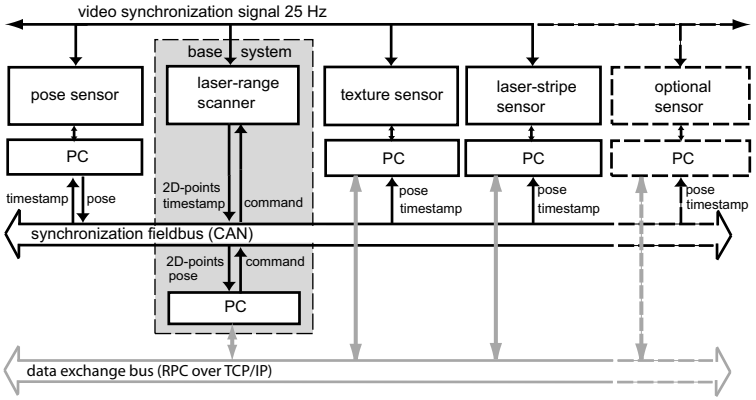


Figure C.2: Synchronization concept of the 3D-Modeller.

sage is sent on the CAN bus. These messages are received by each sensor, which are thus triggered to deliver their data sets accordingly. In addition to the laser-range scanner, the pose sensor sends its data directly via the CAN bus. Every sensor connected to the CAN bus is thus able to receive the current pose in order to reference its local 2-D data sets to the global frame.

Each message on the CAN bus is processed according to its priority. High-priority signals are command and synchronization messages, because their delivery is time critical. The command signals (e.g. *laser_on*, *motor_off*) rarely appear, so they can have the highest priority. The synchronization signals are of lower priority than commands, but higher than the laser-range scanner data. The pose data signal is of lowest priority: due to the timestamp, a delay of this signal relative to the respective data and synchronization message does not provoke false assignments, the merging can be done afterwards. The synchronization concept is presented in Fig. C.2. The CAN bus baudrate is 1 MBit/s. The 2.5-D data sets of the laser-stripe sensor and the images of the texture sensor are merged with the pose data through timestamp identification. The data is available for applications through a unified Remote Procedure Call (RPC) interface ; the application layer has no access to the CAN bus, as it is treated as an internal bus system.

All sensors are to measure synchronously, allowing the implementation of data fusion [1] principles. For this reason, the sensors must not disturb or interfere with each other. Exemplarily, the laser-range scanner's laser line should not be visible in the texture sensor's images when using them simultaneously. A solution is an adjustment of the phase between laser-range scanner and texture sensor, assuring that camera shots are taken while the laser is not visible. It is possible to automatically gain texture information while scanning the object's surface with the laser-range scanner. The laser-range scanner must not be interfered by the laser-stripe sensor. This is avoided by the laser-

(a) Point cloud (b) Mesh (c) Texture

Figure C.3: 3-D modeling: From point cloud to textured model of a human hand.

range scanner's surrounding light compensation feature. The infrared flashes of the tracking system cause noise in the measurements of the laser-range scanner. Knowing the time relation between video pulse and flash appearance allows an exclusion of values measured during the flash period.

The 3-D data of the laser-range scanner and the laser-stripe sensor are acquired and displayed in the same global coordinate system, so together with the timing concept described, the hand-guided 3D-Modeller is capable of real-time data fusion. In Fig. C.3, results for a simultaneous acquisition of surface geometry and corresponding texture are presented.

D

Software Tool: SBP-Simulator

This chapter describes the features of the the Sensor-based Motion Planning (SBP) Simulation Environment, the SBP-Simulator, as developed and applied for the simulations (cf. Chap. 6)and experiments (cf. Chap. 8).

The SBP-Simulator features a graphical user interface (GUI). It not only provides visual monitoring of the entire physical space scene, but functionalities for a maximum simulation control as well. In accordance with the modular structure of the program, the GUI (cf. Fig. D.1) does not need to be displayed during simulation; alternatively, the program can be run via the command line with arguments defining the view planning strategy.

Physical Space Implementation

The physical space is implement as a sphere-tree, constructed with an octree method. Each level of the tree can be accessed by the visualization, collision detection, and view planning module via the control interface. An Unified Modeling Language (UML) diagram of the implementation is depicted in Fig. D.2.

Path Generation

In the *Add Path* tab, a path can be generated between the current position of the robot and any other configuration in the roadmap, provided it is connected to the current guard group (cf. Sec. 6.2.3).

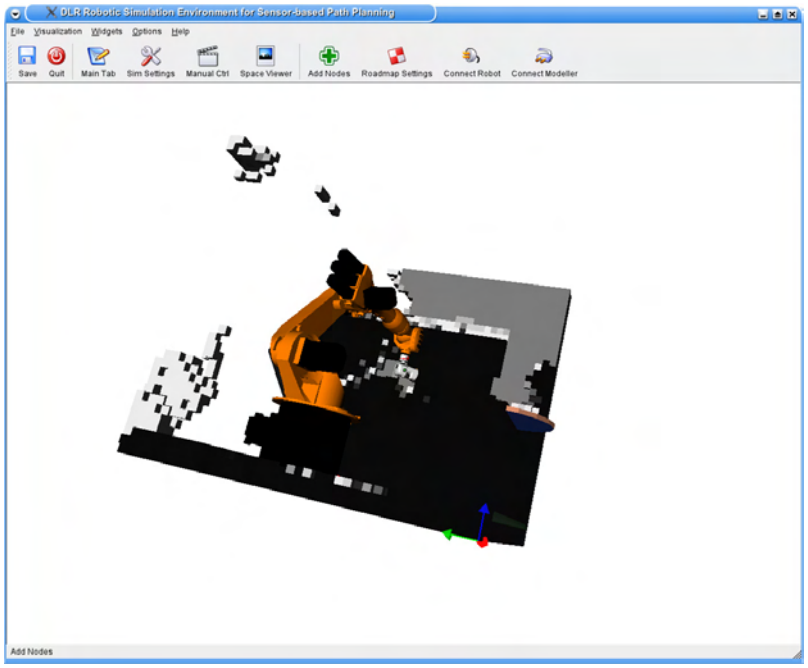


Figure D.1: GUI of the SBP-Simulator.

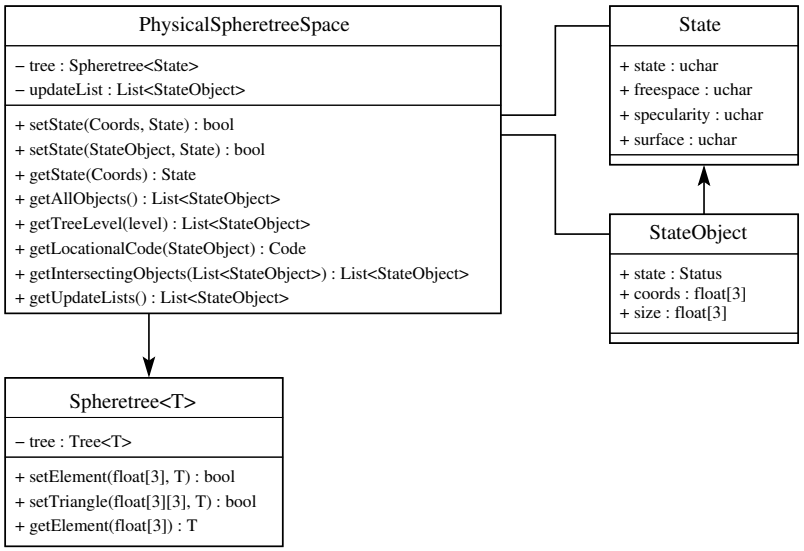


Figure D.2: Data hierarchy for $\mathcal{P}$-space, denoted PSTS, applied in the majority of the simulations.

The *Scan* button, which executes an immediate scan of the physical space from the robot's current configuration using the active sensor, is also located in this tab.

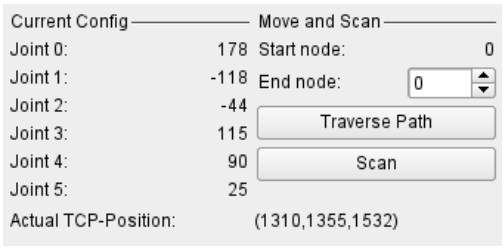


Figure D.3: Manual control dialogue: To the left, the current configuration of the robot is shown. The buttons for executing scans and moving the robot are located on the right.

Robot Animation

A forward kinematics/dynamics solver object, $fwDyn$, is used to position the robot given a vector of joint angles. An inverse kinematics/dynamics solver object, ik, is used to derive inverse kinematic solutions of the robot. The robot moves from configuration A to B by extracting a discretized path between the two configurations. This path is then added into a path buffer located in the kinematics module. The robot's configuration is updated to the next one in the path, provided there is a path in the buffer to traverse. If the home position of the C-Rob model does not correspond to the desired home position, a simple vector of offset angles is applied. All joint angles fed into $fwDyn$ and taken from ik must be appropriately adjusted with this offset vector.

C-Rob is a generic toolbox for modeling robots which cardinally expect link files to have similarly oriented coordinate frames when the robot is in home position. There is, however, an alternative to modifying the coordinate frames of every link. By means of *CrobRotateCoordAxes* objects, the C-Rob kinematic model can be designed to expect links with any desired coordinate frame.

Adding Nodes

A node can be manually added to the robot by explicitly assigning joint values to a configuration through the *Add Node* tab shown in Fig. D.4. Only joint values that are not assigned as *sensor joints* can be assigned (cf section6.2.3). The *sensor joints* are automatically set to the midpoint of their angular range. Upon pressing the *Add Node to Roadmap* button, the program will attempt to add the configuration defined in the fields to the roadmap returning either a message of success or failure. If a node cannot be added to the roadmap, it means that the configuration was either in collision or not within the current *visibility* range of the existing roadmap configurations. Angles are indicated in degrees.

Figure D.4: Tab window for adding nodes. Left: Adding nodes by joint angles. Right: Adding nodes by TCP, F_{sensor}, or FoV to the roadmap.

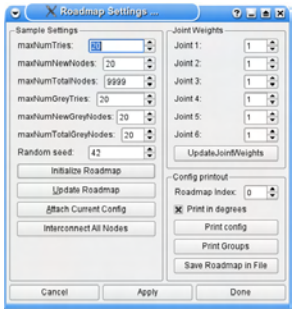


Figure D.5: Roadmap settings tab.

Table D.1: Visibility-based roadmap parameter description

Parameter	Variable	Description
maxNumTries	n_{rmtry}	Maximal number of consecutive failed attempts to add a node to the roadmap before stopping roadmap update.
maxNumNewNodes	n_{rm}	Maximal number of new nodes to add before stopping roadmap update.
maxNumTotalNodes	n_{maxrm}	Maximal capacity of the roadmap.
maxNumGreyTries	n_{unktry}	Maximal number of consecutive failed attempts to place a grey node before stopping grey node placement.
maxNumNewGreyNodes	n_{unk}	Maximal number of new grey nodes placed before stopping grey node placement.
maxNumTotalGreyNodes	n_{maxunk}	Maximal capacity of stored grey nodes.
Random Seed		Random seed affects generation of random nodes. To vary roadmap configurations, select a different seed.

Via inverse kinematics, a configuration can be added to the roadmap by specifying a TCP in world-coordinates. Using the interface shown in Fig. D.4 a position can be specified either as TCP, sensor origin, or sensor FoV midpoint. The direction is specified as a vector in Cartesian space, however, its magnitude is irrelevant as the vector is internally normalized.

Roadmap Settings

The visibility roadmap parameters can be set in the *Roadmap Settings* tab shown in Fig. D.5. These parameters, described in Table D.1, are mostly termination criteria for the roadmap and grey node update/expansion functions.

Pressing the *Update Roadmap* button prompts an expansion of the roadmap by the Visib-PRM method. The *Initialize Roadmap* button performs the same action after deleting all of the nodes in the roadmap first.

Data Export and Scene Capture

Simulation output data can be automatically recorded by means of scene captures and data exportation. Inventor scene graphs can be captured manually in the *Display Options* section shown in Fig. D.6 where pressing *Take snapshot* will capture the scene currently displayed in the viewer as an indexed file called 'rootSceneGraph*X*.iv' where *X* is the current number of snapshots of the entire scene that have been taken since the program was started. The exported file can then be read by an inventor file viewer (cf. Fig. D.7).

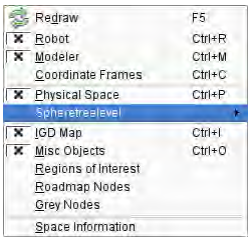


Figure D.6: Display settings.

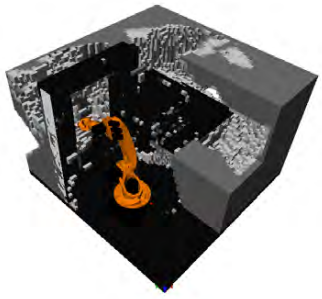


Figure D.7: Captured root scene (without roadmap nodes and grey nodes).

Settings for automatic captures can be defined in the *Capture Scene* tab. The user can specify what to export (i.e. IG Map, pSpace, and/or entire root scene) and how often to export it.

Raw data is also exported in the form of a ';'-delimited ASCII file called *timeStamp.log* indicating simulation iteration counts, view planning strategies, selected view node indices, and iteration times. This file is overwritten every time the program is restarted.

Configuration Space 2-D Map

The configuration space rendering module equips the program with the capability of creating and rendering 2-D slice maps in the robot's configuration space. Naturally, the configuration space can, and usually does, have a dimension greater than two, but it is difficult to visualize a 3D space and nearly impossible to visualize a higher dimensional one.

An example of the map is shown in Fig. D.8. The joints defining the two axes of the map (marked by checkboxes) are swept from -180 to +180 degrees, while the other joints are locked at user-defined values. Blue represents regions of configurations in collision, white represents regions of free configurations, and green represents regions of config-

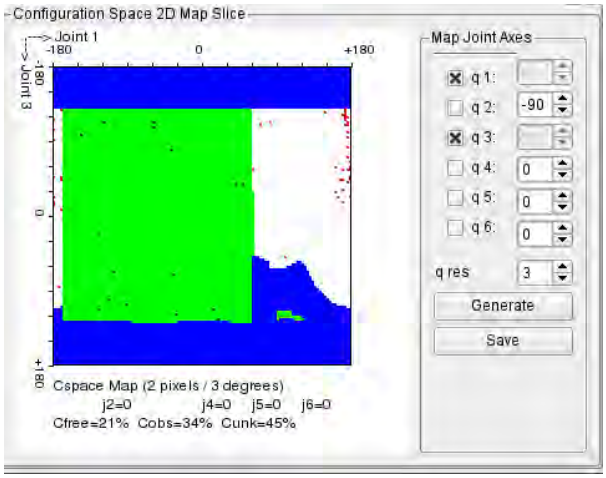


Figure D.8: $\mathcal{C}$-space 2-D map. Red dots indicate roadmap nodes, black dots depict grey nodes. White areas represent $\mathcal{C}_{free}$, dark blue areas describe $\mathcal{C}_{occ}$, including the joint limits. The green areas depict $\mathcal{C}_{unk}$.

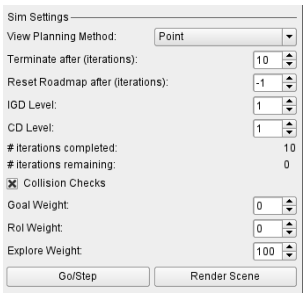


Figure D.9: Simulation settings GUI.

urations that intersect with unknown space.

Simulation Mode and View Planning Strategy Selection

From the *Simulation Settings* section, the user can select any of the view planning strategies integrated into the program. The simulation can be run in step mode (single iterations) or continuous mode, in which the number of iterations can be specified. The *Go/Step* button initiates the simulation and. In continuous mode it turns into a *Stop* button during simulation so that it can be interrupted in between iterations. Upon pressing *Stop*, the program will complete the current iteration before terminating the simulation process.

Command Line Execution

The simulation program can also be executed via the command line without generating a GUI interface. Fewer resources are consumed at the expense of restricted simulation control. The parameters of the simulation must be entered as arguments into the command line. Examples are shown in Tab. D.2.

Table D.2: SBP-Simulator command line options. Typing *sbp-simulator -h* will prompt the help menu.)

Command	*Description*
sbp-simulator	Execute program with GUI using Kuka robot (default)
sbp-simulator -nogui brs	Execute program without GUI using Kuka robot and begin simulation using Beam Roadmap Sweep view planning strategy.
sbp-simulator -nogui p -robot kr16	Execute program without GUI using the KR16 robot and begin simulation using Point view planning strategy.
sbp-simulator -nogui p i w_{expl} w_{goal} w_{RoI} l_{IG} l_{CD} -connect	Execute program without GUI using the simulation using Point with i iterations and the corresponding weights for the task w_{task}. The IG level and CD level is provided, the robot is connected.

XIRP Client

A visualization of the current state of the environment is not available in console mode. Additionally, the SBP-Simulator can be executed on a system without monitor. In order to visualize the $\mathcal{P}$-space a XIRP Client is developed. A screenshot of the client is depicted in Fig. D.10. The XIRP client is directly connected to the control interface as visualized in Fig. D.11.

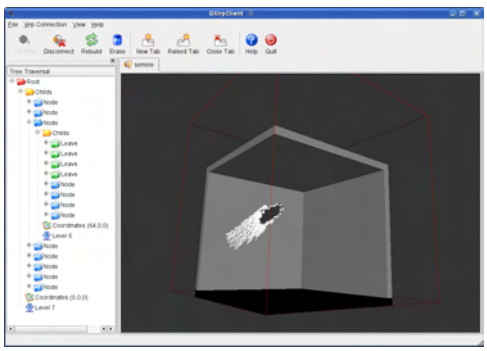


Figure D.10: XIRP Client.

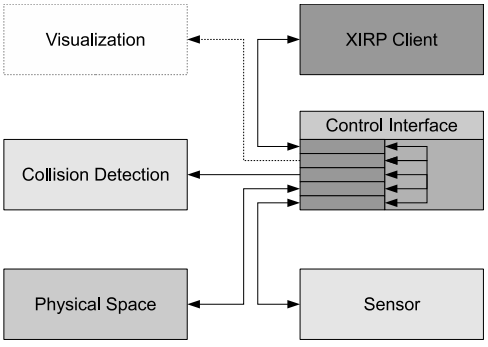


Figure D.11: Integration of the XIRP Client into the control interface.

Robot Sensor Interface

The module rsiCan is a Robot Sensor Interface (RSI) object with 6 inputs and 7 outputs. The inputs should be linked with the current cartesian robot position from object `ST_ACTPOS`. This current position is transmitted via CAN. Outputs 1 to 6 are reserved for future use and will contain the pose obtained from a tracking system via CAN. This has not been implemented yet. Output 7 contains the timestamp received from the 3D-Modeller as received via CAN. Note that the unit of all inputs and outputs is `'RSIUNIT_No'`.

The module requires that at least RSI V2.01 is installed on the KRC. The binary of the RSI module *rsiCan.out* must be copied to `C:\KRC\ROBOTER\Drivers\`. Then, the KRL library that makes the module usable, e.g. *rsiCanLib.src* and *rsiCanLib.dat*, must be transferred to `C:\KRC\ROBOTER\KRC\R1\TP\RSI\`. The file *iosys.ini* must be edited in `C:\KRC\ROBOTER\INIT\`: in order to automatically load the driver at robot start-up: a line `'RSICAN=XXX,rsiCanInit,rsiCan.out'` with `XXX` being a number not used, is inserted and an empty section `'[RSICAN]'` must be added. Finally, the robot must be restarted.

E

Applications Beyond Exploration: Medical Robotics and Cultural Heritage Preservation

In robotics, depth information is used to avoid collisions, to navigate through an environment, or to plan a grasp of an object. Texture information is used to self locate the robot in its environment as well as to find and identify objects. The concept developed in this thesis allows application in other areas: The measured data has to be processed fast, such that the robot can use the results immediately. In the following, two applications of the 3D-Modeller currently beyond exploration are detailed.

E.1 Medical Applications

Experimental evaluations in medical applications of the 3D-Modeller show promising results.

The LRS operates robustly in the challenging lighting conditions of Operation Rooms (OR). The task performed is the registration of patients. A hand-guided application is preferred: the surgeon [83, 85] acquires a surface scan, which is used to align the pre-operative patient data (CT surface model) with the OR situation. The target area is head surgery. Additionally, exploration, i.e. view planning in order to reduce model error for the registrations, must be researched. The proposed view points can either be augmented to the surgeon, or a medical robot can perform a measurement automatically.

At DLR, the medical robot Kinemedic has been developed. This 7-DoF robot is to be used as a supporting system for the surgeon, providing haptic feedback to the surgeon in minimally invasive surgery. Range measurements outside the human body are not feasible to determine the state of the physical space, however, skin sensors and torque sensors of the robot, usually applied for haptic feedback to the surgeon, can be used in order to detect $\mathcal{P}_{free}$. Task-specific view planning for endoscopes, similar to SCS, is beneficial for the surgeon in order to provide consistent augmentation of the surgery-related information, i.e. compute optimal views of the target area in order to reduce model error.

E.2 Cultural Heritage Preservation

Various types of sensors and methods are needed for modeling different sized cultural objects. They range from small objects, like bust and statues, to medium-scale objects, like rooms and building interiors, and large terrains. Combining different sensors allows for acquiring any object with different levels of detail, e.g. fast digitization of a large object at medium resolution and refining some parts with higher accuracy afterwards. Acquisition and data processing is very time consuming, if done manually. Again, methods from the robotic field may help automating both, acquisition and modeling. Common for all scales is the processing, i.e. the modeling and model refinement. Beneficial in starting off from robotics is the diversity of robotic environments: the methods applied should not be specific for one scale. Therefore, the generic methods and sensor concepts to fusion of color and geometry information in an unique 3-D model are easily extended to all scales in cultural heritage.

In order to choose the right sensor system for acquisition, the approach is categorized into three scale levels:

1. Small-scale: Sensing distance of $0.05m$ to $2m$, resolution maximally $0.001m$ to $0.005m$.

2. Medium-scale: Sensing distance of $2m$ to $50m$, resolution $0.05m$ to $0.1m$.

3. Large-scale: Sensing distance of larger than $50m$, resolution from $0.2m$ up to $1m$.

The sensor systems satisfying this scale range definition as well as their application is presented in the following. Details on the methods are described in [72, 104].

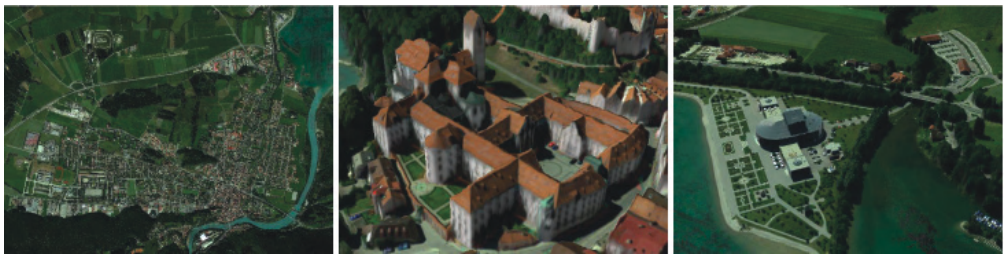


Figure E.1: Visualization of a reconstruction of Fuessen. The images cover an area of 24 km2 and have a ground resolution of 15 cm/pixel, leading to a DEM and ortho-image consisting of 1 billion height values and pixels.

Small-Scale Models The 3D-Modeller is applied for modeling small busts and objects in the domain of cultural heritage. In this field, robots are currently not used in indoor modeling. Therefore, an operator sweeps the system manually over the surface of the object, and the 3-D model is reconstructed simultaneously by the surface generation algorithm. Immediate visual feedback helps the user to completely digitize the object. Additionally, live images from the texture sensor are integrated into the visual feedback.

Medium-Scale Models In the medium scale, the digitization of historic buildings is performed. The focus is set on the historic site of castle Neuschwanstein near Fuessen, Germany, where maps and views of approx. 450 rooms of the castle have been generated. Additionally, 60 rooms have been reconstructed in a 3-D model, some of them colored. Data acquisition was performed using the *Z+F Imager* [89, 54] and the *DLR panoramic camera*.

Large-Scale Models The SGM stereo method [70] has been used for processing several huge areas from preprocessed High Resolution Stereo Camera (HRSC) images [184, 134]. Side textures of buildings and other objects are taken from the forward and backward looking stereo images. The whole process of stereo matching, Digital Elevation Model (DEM) generation, ortho-imaging, side texture creation and visualization is fully automatic. Fig. E.1 shows visualizations of the city Fuessen, which were recorded in 10 parallel, partly overlapping airborne acquisitions from an altitude of 1750m.

The DEM models can be used for coarse navigation planning, enabling view planning for mobile robots.

Bibliography

[1] M. Abidi and R. Gonzales. *Data Fusion in Robotics and Machine Vision*. Academic Press, 1992. ISBN 0-12-042120-8.

[2] S. Abrams and P. Allen. Computing swept volumes. *The Journal of Visualization and Computer Animation*, pages 69–82, 2000.

[3] S. Abrams, P. Allen, and K. Tarabanis. Computing camera viewpoints in an active robot work cell. *The International Journal of Robotics Research*, 18(3):267–285, February 1999.

[4] E. U. Acar, H. Choset, A. A. Rizzi, P. N. Atkar, and D. Hull. Morse decompositions for coverage tasks. *International Journal of Robotics Research*, 21(4):331–344, April 2002.

[5] A. Agathos and R. B. Fisher. Colour texture fusion of multiple range images. In *Proc. of 4th Int. Conf. on 3-D Digital Imaging and Modeling (3DIM 2003)*, Banff, Canada, 2003.

[6] J. M. Ahuactzin and A. Portilla. A basic algorithm and data structure for sensor-based path planning in unknown environments. In *International Conference on Intelligent Robots and Systems IROS*, Takamatsu, Japan, 2000.

[7] P. Allen, S. Feiner, A. Troccoli, H. Benko, E. Ishak, and B. Smith. Seeing into the past: Creating a 3d modeling pipeline for archaeological visualization. In *3D Data Processing, Visualization and Transmission Symposium*, Salonik, Greece, Sep 6–9 2004.

[8] P. Allen, I. Stamos, A. Gueorguiev, E. Gold, and P. Blaer. Avenue: Automated site modeling in urban environments. In *International Conference on 3D Digital Imaging and Modeling (3DIM)*, pages 357–364., Quebec City, Quebec, Canada, 2001.

[9] P. Allen, I. Stamos, A. Troccoli, B. Smith, M. Leordeanu, and Y. C. Hsu. 3d modeling of historic sites using range and image data. In *ICRA 2003*, Taipai, Taiwan, May 11-16 2003.

[10] T. Arbel and F. P. Ferrie. Entropy-based gaze planning. *Image and Vision Computing*, 19:779–786, 2001.

[11] H. Baltzakis, A. Argyros, and P. Trahanias. Fusion of laser and visual data for robot motion planning and collision avoidance. *Machine Vision and Applications*, 15:92–100, 2003.

[12] J. E. Banta, L. M. Wong, C. Dumont, and M. A. Abidi. A next-best-view system for autonomous 3-d object reconstruction. *IEEE Transactions on Systems, Man and Cybernetics*, 3(5):589–598, September 2000.

[13] J. Barraquand and J. C. Latombe. Robot motion planning: A distributed representation approach. *International Journal on Robotics Research*, 10(6):628–649, 1991.

[14] F. Bernardini and H. Rushmeier. The 3d model acquisition pipeline. *Computer Graphics Forum*, 21(2):149, June 2002.

[15] P. J. Besl. Active, optical range imaging sensors. *Machine Vision and Applications*, 1:127–152, 1988.

[16] J. Beyerer. *Informationsfusion in der Mess- und Sensortechnik*. J. Beyerer (Hrsg.) and F. Puente León (Hrsg.) and K.-D. Sommer (Hrsg.), 2006.

[17] P. Blaer and P. Allen. Two stage view planning for large-scale site modeling. In *Intl. Symposium on 3D Data Processing, Visualization and Transmission*, June 2006.

[18] P. Blaer and P. Allen. View planning for automated site modeling. In *IEEE International Conference on Robotics and Automation (ICRA)*, Orlando, Florida, U.S.A., May 2006.

[19] T. Bodenmueller and G. Hirzinger. Online surface reconstruction from unorganized 3d-points for the DLR hand-guided scanner system. In *2nd Symposium on 3D Data Processing, Visualization, Transmission*, Salonika, Greece, Sep 6–9 2004.

[20] T. Bodenmueller, W. Sepp, M. Suppa, and G. Hirzinger. Tackling multisensory 3d data acquisition and fusion. In *To appear 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, San Diego, U.S.A., 2007.

[21] J. Borenstein and Y. Koren. Real-time obstacle avoidance for fast mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(5):1179–1187, 1989.

[22] A. Bottino and A. Laurentini. What's next? an interactive next best view approach. *Pattern Recognition*, 39(1):126–132, January 2006.

[23] G. Bradshaw. *Bounding Volume Hierarchies for Level-of-Detail Collision Handling*. PhD thesis, Trinity College Dublin, Dept. of Computer Science, 2002.

[24] G. Bradshaw and C. O'Sullivan. Adaptive medial-axis approximation for sphere-tree construction. *ACM TOG*, 2002.

[25] G. Bradshaw and C. O'Sullivan. Sphere-tree construction using medial-axis approximation. *ACM SIGGRAPH Symposium on Computer Animation SCA*, 2002.

[26] O. Brook and O. Khatib. Real-time obstacle avoidance and motion coordination in a multi-robot workcell. In *International Symposium on Assembly and Task Planning*, pages 274–279, Porto, Portugal, July 1999.

[27] O. Brook and O. Khatib. Real-time replanning in high-dimensional configuration spaces using sets of homotopic paths. In *International Conference on Robotics and Automation (ICRA)*, pages 550–555, 2000.

[28] L. G. Brown. A survey of image registration techniques. *ACM Computer Survey*, 24(4):325–376, 1992.

[29] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Estimating the absolute position of a mobile robot using position probability grids. In *Proceeding of the 14th National Conference on Artificial Intelligence (AAAI-96)*, 1996.

[30] W. Burgard, M. Moors, C. Stachniss, and F. Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3):376–378, 2005.

[31] R. Burger and M. Suppa. Anbindung und Evaluierung von Sensorik zur Erzeugung von Umgebungsmodellen in Robotersteuerungen. Master's thesis, FH Muenchen, 2007.

[32] B. Burns and O. Brock. Toward optimal configuration space sampling. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[33] B. Burns and O. Brock. Sampling-based motion planning with sensing uncertainty. In *Proceedings IEEE International Conference on Robotics and Automation*, Rome, Italy, April 2007.

[34] D. Burschka and G. D. Hager. V-GPS(SLAM): – Vision-Based Inertial System for Mobile Robots. In *Proc. of ICRA*, pages 409–415, April 2004.

[35] S. Y. Chen and Y. F. Li. Automatic sensor placement for model-based robot vision. *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics*, 34(1):393–408, February 2004.

[36] S. Y. Chen and Y. F. Li. Vision sensor planning for 3-d model acquisition. *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics*, 35(5):894–904, October 2005.

[37] H. Choset and J. Burdick. Sensor-based exploration: The hierarchical generalized voronoi graph. *International Journal of Robotics Research*, 19(2):96–125, February 2000.

[38] H. Choset, S. Walker, K. Eiamsa-Ard, and J. Burdick. Sensor-based exploration: Incremental construction of the hierarchical generalized voronoi graph. *International Journal of Robotics Research*, 19(2):126–148, February 2000.

[39] O. Cohen, Y. Edan, and E. Schechtman. Statistical evaluation method for comparing grid map based sensor fusion algorithms. *International Journal of Robotics Research*, 25(2):117–133, February 2006.

[40] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, 1996.

[41] A. Dempster. A generalization of bayesian inference. *Journal of Royal Statistical Society*, 30:205–247, 1968.

[42] M. Desbrun, M. Meyer, P. Schroeder, and A. H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *SIGGRAPH 99*, pages 317–324, 1999.

[43] S. Dietl and M. Suppa. Integration eines Robotergefhrten Hand-Auge-Systems zur Umgebungsexploration und Vergleich von Explorationsverfahren. Master's thesis, TU Muenchen, 2006.

[44] K. Dietmayer. Evidenztheorie: Ein Vergleich zwischen Bayes- und Dempster-Shafer-Methoden. In *Informationsfusion in der Mess- und Sensortechnik*. J. Beyerer (Hrsg.) and F. Puente León (Hrsg.) and K.-D. Sommer (Hrsg.), 2006.

[45] E. W. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[46] L. Dorst, M. Van Lambalge, and F. Voorbraak. *Reasoning with Uncertainty in Robotics RUR*. Lecture Notes in Computing Science 1093. Springer, 1995.

[47] A. Elfes. *Occupancy Grids: A Probabilistic Framework for Mobile Robot Perception and Navigation*. PhD thesis, Electrical and Computer Engineering Department/Robotics Institute, Carnegie Mellon University, 1989.

[48] M. Fernandez, K. Gupta, and J.C. Fraile. Simultaneous path planning and exploration for manipulators with eye and skin sensors. In *Proceedings. 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 914 – 919, 2003.

[49] R. Fisher, A. Fitzgibbon, A. Gionis, M. Wright, and D. Eggert. A hand-held optical surface scanner for environmental modeling and virtual reality. In *Proceedings Virtual Reality World, Stuttgart*, pages 13–15, Stuttgart, Germany, 1996.

[50] A. W. Fitzgibbon, G. Cross, and A. Zisserman. Automatic 3d model construction for turn-table sequences. In *SMILE*, pages 155–170, 1998.

[51] W. Foerstner. A feature based correspondence algorithm for image processing. *International Archives of Photogrammetry and Remote Sensing*, 26:150–166, 1986.

[52] L. Freda, F. Loiudice, and G. Oriolo. A randomized method for integrated exploration. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, Bejing, China, 2006.

[53] L. Freda and G. Oriolo. Frontier-based probabilistic strategies for sensor-based exploration. In *IEEE International Conference on Robotics and Automation (ICRA)*, Barcelona, Spain, 2006.

[54] C. Froehlich. Aktive Erzeugung korrespondierender Tiefen- und Reflektivitätsbilder und ihre Nutzung zur Umgebungserfassung. *Dissertation*, 1. Auflage, 1996.

[55] C. Frueh and A. Zakhor. An automated method for large-scale, ground-based city model acquisition. *International Journal on Computer Vision*, 60(1):5–24, 2004.

[56] F. Gambino, G. Oriolo, and G. Ulivi. A comparison of three uncertainty calculus techniques for ultrasonic map building. In *SPIE*, volume 2761, pages 249–260, 1996.

[57] E. Gilbert, D. Johnson, and S. Sathiya Keerthi. A fast procedure for computing the distance between complex objects in three dimensional space. *IEEE Journal of Robotics and Automation*, 4(2):193–203, 1988.

[58] T. Gockel, J. Ahlmann, P. Azad, and R. Dillmann. 3d vision sensing for grasp planning: A new, robust and affordable structured light approach. In *International Conference on Robotics and Automation (ICRA)*, 2005.

[59] H. H. González-Baños and J.-C. Latombe. A randomized art gallery algorithm for sensor placement. In *Proc. 17th ACM Symp. on Computational Geometry (SoCG)*, pages 232–240, 2001.

[60] H. H. González-Baños and J.-C. Latombe. Navigation strategies for exploring indoor environments. *Robotics Research*, 21(10–11):829–848, October-November 2002.

[61] F. Hacker, J. Dietrich, and G. Hirzinger. A laser-triangulation based miniaturized 2-d range-scanner as integral part of a multisensory robot-gripper. In *EOS Topical Meeting on Optoelectronic Distance/Displacement Measurements and Applications*, Nantes, France, 1997.

[62] D. Haehnel, W. Burgard, and S. Thrun. Learning compact 3d models of indoor and outdoor environments with a mobile robot. In *Robotics and Autonomous Systems*, volume 44, pages 15–27, 2003.

[63] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th ALVEY Vision Conference*, pages 147–151, 1988.

[64] P. Heiligensetzer. *Sichere Mensch-Roboter Kooperation durch Fusion haptischer und kapazitiver Sensorik*. Shaker Verlag GmbH, 2003.

[65] B. Heimann, W. Gerth, and K. Popp. *Mechatronik: Komponenten - Methoden - Beispiele*. Fachbuchverlag Leipzig im Carl Hanser Verlag, 1998.

[66] U. Hillenbrand. Consistent parameter clustering: Definition and analysis. *Pattern Reconition Letters*, 2007.

[67] A. Hilton and J. Illingworth. Geometric fusion for a hand-held 3d sensor. In *Machine Vision and Applications*, pages 44–51, Springer-Verlag, 2000.

[68] H. Hirschmueller. Improvements in real-time correlation-based stereo vision. In *IEEE Workshop on stereo on multi-baseline vision, IJCV*, 2001.

[69] H. Hirschmueller. Real-time map building from a stereo camera under unconstrained 3d motion. In *Faculty Research Conference, Faculty of Computing Science and Engineering, De Montfort University, Leicester, UK*, 2003.

[70] H. Hirschmueller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR)*, San Diego, USA, June 2005.

[71] H. Hirschmueller, P. R. Innocent, and J. M. Garibaldi. Real-time correlation-based stereo vision with reduced border errors. *IJCV*, 47(1/2/3):229–246, April-June 2002.

[72] G. Hirzinger, T. Bodenmueller, H. Hirschmueller, R. Liu, W. Sepp, M. Suppa, T. Abmayr, and B. Strackenbrock. Photo-realistic 3d-modelling - from robotics perception towards cultural heritage.

In *International Workshop on Recording, Modelling and Visualization of Cultural Heritage*, Ascona, Switzerland, 22–27 May 2005.

[73] R. Hoepler and M. Otter. A versatile c++ toolbox for model based, real time control systems of robotic manipulators. In *International Conference on Intelligent Robots and Systems IEEE 2001, Maui, Hawaii, USA*, 2001.

[74] J. Honerkamp. *Stochastische Dynamische Systeme*. VCH Verlagsgesellschaft, 1990.

[75] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Computer Graphics (SIGGRAPH)*, volume 26, pages 19–26, 1992.

[76] D. Hsu, J.-C. Latombe, and H. Kurniawati. On the probabilistic foundations of probabilistic roadmap planning. In *12th Int. Symposium on Robotics Research*, San Francisco, October 2005.

[77] D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. In *In Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 2719–2726, 1997.

[78] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transaction on Robotics and Automation*, 12(4):566–580, August 1996.

[79] S. Kielhoefer and M. Suppa. Fehleranalyse und Modellierung eines 3-D Laserscansystems. Master's thesis, TU München, 2003.

[80] K. Klein and V. Sequiera. View planning for the 3d modelling of real world scenes. In *International Conference on Intelligent Robots and Systems (IROS)*, 2000.

[81] J. Ko, B. Steward, D. Fox, K. Konolige, and B. Limketkai. A practical, decision-theoretic approach to multi-robot mapping and exploration. In *Proc. of IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2003.

[82] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions using graph cuts. In *International Conference for Computer Vision*, volume 2, pages 508–515, 2001.

[83] R. Konietschke, A. Busam, T. Bodenmueller, T. Ortmaier, J. Wiechnik, M. Suppa, T. Welzel, G. Eggers, G. Hirzinger, and R. Marmulla. Potential, limitations and challenges of markerless registration with the DLR 3D-Modeller in medical applications. In *Proceedings of the 21st International Congress and Exhibition*

of Computer Assisted Radiology and Surgery, Berlin, Germany, June 2007.

[84] K. Konolige. Improved occupancy grids for map building. *Autonomous Robots, Kluwer Academic Press*, 4:351–367, 1997.

[85] W. Korb, T. Bodenmueller, G. Eggers, T. Ortmaier, M. Schneberger, M. Suppa, J. Wiechnik, R. Marmulla, and S. Hassfeld. Surface-based image-to-patient-registration using a hand-guided laser-range scanner system. In *CARS 2004; Computer Assisted Radiology and Surgery, 18th International Congress and Exhibition, Chicago, USA, June 23*, page 1328, 2004.

[86] E. Kruse, R. Gutsche, and F. M. Wahl. Efficient, iterative, sensor based 3-d map building using rating functions in configuration space. In *IEEE International Conference on Robotics and Automation (ICRA)*, Minneapoliy, Minnesota, April 1996.

[87] K. Kuehnlenz, M. Bachmayer, and M. Buss. A multi-focal high-performance vision system. In *Proc. of Int. Conference on Robotics and Automation ICRA*, Orlando, U.S.A., 2006.

[88] J. J. Kuffner and S. M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *In Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 995–1001, 2000.

[89] D. Langer, J. Hancook, M. Martial Hebert, E. Hoffmann, M. Mettenleiter, and C. Froehlich. Active laser radar for high-performance measurements. In *IEEE Robotics and Automation ICRA 98*, Leuven, Belgium, May 16-21 1998.

[90] A. Laurentini. The visual hull of curved objects. In *Proc. IEEE International Conference on Computer Vision (ICCV) (1)*, pages 356–361, 1999.

[91] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[92] S. M. LaValle and J. J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In *In Proceedings of International Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages SA45–SA59, 2000.

[93] J. Yeong Lee and H. Choset. Sensor-based exploration for convex bodies: A new roadmap for a convex-shaped robot. *IEEE Transactions on Robotics*, 21(2):240–247, April 2005.

[94] B. R. Leffler, M. L. Littman, A. L. Strehl, and T. J. Walsh. Efficient exploration with latent structure. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[95] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital michelangelo project: 3d scanning of large statues. In *Proc. SIGGRAPH 2000*, 2000.

[96] Y. F. Li and Z. G. Liu. Information entropy-based viewpoint planning for 3-d object reconstruction. *IEEE Transactions on Robotics*, 21(3):324–337, June 2005.

[97] P. Liepa. Filling holes in meshes. In *Eurographics /ACM SIGGRAPH Symposium Geometry Processing*, pages 200–205, May 16-21 2003.

[98] M. Lin and S. Gottschalk. Collision detection between geometric models: A survey. In *Proceedings of IMA Conference on Mathematics of Surfaces*, 1998.

[99] M. Lin, D. Manocha, J. Cohen, and S. Gottschalk. Collision detection: Algorithms and applications. In Jean-Paul Laumond and M. Overmars (invited submission), editors, *Proceedings of Algorithms for Robotics Motion and Manipulation*, pages 129–142. A.K. Peters, 1996.

[100] M. Lin and D. Minocha. *Handbook of Discrete and Computational Geometry*, chapter 35 Collision and proximity queries. J.E. Goodman and J. O'Rourke (Editors), CRC Press, 2004.

[101] M. C. Lin and J. Canny. A fast algorithm for incremental distance calculation. In *IEEE Conference on Robotics and Automation (ICRA)*, pages 1008–1014, 1991.

[102] M. C. Lin, D. Manocha, and J. Canny. Fast contact determination in dynamic environments. In *IEEE Conference on Robotics and Automation (ICRA)*, pages 602–609, 1994.

[103] C. Liska and R. Sablatnig. Adaptive 3d acquisition using laser light. In Tomas Svoboda, editor, *Czech Pattern Recognition Workshop 2000*. Czech Pattern Recognition Society, 2000.

[104] R. Liu, M. Suppa, G. Hirzinger, and D. Burschka. Modelling the world in real time. In *Proceedings of the 8th Conference on Optical 3-D Measurement Techniques*, Zurich, Switzerland, July 9–12 2007.

[105] Y. Ma, S. Soatto, J. Košecká, and S. Shankar Sastry. *An Invitation to 3-D Vision From Images to Geometric Models*. Springer, 2004.

[106] L. Matthies and A. Elfes. Integration of sonar and stereo range data using a grid-based representation. In *IEEE International Conference on Robotics and Automation (ICRA)*, Philadelphia, PA, USA, April 1988.

[107] L. Matthies and P. Grandjean. Stochastic performance modeling and evaluation of obstacle detectability with imaging range sensors. *T-RA*, 10:783–792, 1994.

[108] J. Maver and R. Bajcsy. Occlusions as a guide for planning the next view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5), May 1993.

[109] E. Mazer, J. M. Ahuactzin, and P. Bessière. The ariadne's clew algorithm. In *Journal of Artificial Intelligence Research*, pages 295–316, 9, 1998.

[110] G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley Series in Probability and Statistics. John Wiley and Sons, 1997.

[111] L. McMillan. Computing visibility without depth. Tr95-047, Department of Computer Science, University of North Carolina, Chapel Hill, Sitterson Hall, Chapel Hill, NC 27599, 1995.

[112] M. A. Morales, R. Pearce, and N. M. Amato. Metrics for analyzing the evolution of c-space models. In *International Conference on Robotics and Automation (ICRA)*, Orlando, Florida, U.S.A., 2006. IEEE.

[113] H. P. Moravec. Robot spatial perception by stereoscopic vision and 3d-evidence grids. Technical Report CMU-RI-TR-96-34, Robotics Institute, Carnegie Mellon University, 1996.

[114] D. Murray and J. Little. Using real-time stereo vision for mobile robot navigation. In *IEEE Transaction on System, Man and Cybernetics*, volume 25, pages 10–20, January 2000.

[115] C. Nissoux, T. Siméon, and J.-P. Laumond. Visibility based probabilistic roadmaps. In *IEEE International Conference on Intelligent Robots and Systems*, 1999.

[116] M. Ordaz, E. Whittenberger, D. Waagen, and D. Hulsey. A statistical analysis of 3d structure tensor features generated from ladar imagery. In *Proceedings of the SPIE: Automatic Target Recognition XVI.*, volume 6234 of *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, June 2006.

[117] G. Oriolo, G. Ulivi, and M. Vendittelli. Real-time map building and navigation for autonomous robots in unknown environments. In *IEEE Transactions on Systems, Man, and Cybernetics*, pages 100–135, 1999.

[118] G. Oriolo, M. Vendittellia, L. Freda, and G. Troso. The srt method: Randomized strategies for exploration. In *Proceedings*

of the IEEE International Conference on Robotics and Automation, New Orleans, LA, U.S.A., April 2004.

[119] J. O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, 1987.

[120] J. O'Rourke. *Computational Geometry in C (Second Edition)*. Cambridge University Press, 1998.

[121] C. Ott, O. Eiberger, W. Friedl, B. Baeuml, U. Hillenbrand, C. Borst, A. Albu-Schaeffer, B. Brunner, H. Hirschmueller, S. Kielhoefer, R. Konietschke, M. Suppa, T. Wimboeck, F. Zacharias, and G. Hirzinger. A humanoid two-arm system for dexterous manipulation. In *HUMANOIDS'06*, Genoa, Italy, 2006.

[122] N. Ott. *Unsicherheit, Unschärfe und rationales Entscheiden - Die Anwendung von Fuzzy Methoden in der Entscheidungstheorie*. Physica Verlag, 2001.

[123] D. Pagac, E.M. Nebot, and H. Durrant-Whyte. *Reasoning with Uncertainty in Robotics RUR*, chapter An Evidential Approach to Probabilistic Map Building, pages 164–170. Lecture Notes in Computing Science 1093. Dorst, L. and Van Lambalge, M. and Voorbraak, F. (Editors), Springer, 1995.

[124] R. Pito. A sensor-based solution to the next best view problem. In *International Conference on Pattern Recogniction (ICPR)*, Vienna, Austria, 1996.

[125] R. Pito and R Bajcsy. A solution to the next best view problem for automated CAD model acquisition of free-form objects using range cameras. In *Symposium on Intelligent Systems and Advanced Manufacturing (SPIE)*, Philadelphia, USA, 1995.

[126] M. K. Reed and P. K. Allen. A robotic system for 3d-model acquisition from multiple range images. In *Proceedings Intl. Conference on Robotics and Automation (ICRA)*, pages 2509–2514, 1997.

[127] M. K. Reed and P. K. Allen. 3-d modeling from range imagery: An incremental method with a planning component. *Image and Vision Computing*, 17:99–111, 1999.

[128] E. Rimon and J. F. Canny. Construction of c-space roadmaps from local sensory data: What should the sensors look for? In *IEEE Conference on Robotics and Automation ICRA*, 1994.

[129] S. Rusinkiewicz and M. Levoy. Qsplat: A multiresolution point rendering system for large meshes. In *SIGGRAPH*, 2000.

[130] H. Samet. *Applications of Spatial Data Structures*. Addison-Wesley Publishing Company, Inc., 1990.

[131] N. Sawasaki, M. Nakao, Y. Yamamoto, and K. Okabayashi. Embedded vision system for mobile robot navigation. In *Proc. of Int. Conference on Robotics and Automation ICRA*, Orlando, U.S.A., 2006.

[132] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1/2/3):7–42, April-June 2002.

[133] G. Schaufler, J. Dorsey, X. Decoret, and F. X. Sillion. Conservative volumetric visibility with occluder fusion. In *SIGGRAPH*, 2000.

[134] F. Scholten, K. Gwinner, and F. Wewel. Angewandte digitale photogrammetrie mit der hrsc-a. In *Photogrammetrie, Fernerkundung, Geoinformation*, volume 5, pages 317–332, Stuttgart, Germany, 2002. E. Schweizerbart Verlagsbuchhandlung.

[135] W. R. Scott, G. Roth, and J.-F. Rivest. View planning with a registration constraint. In *3D Imaging and Modeling Conference (3DIM)*, Québec City, Québec, Canada, 2001.

[136] W. R. Scott, G. Roth, and J.-F. Rivest. View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys*, 35(1):64–96, March 2003.

[137] W. Sepp, S. Fuchs, and G. Hirzinger. Hierarchical featureless tracking for position-based 6-dof visual servoing. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, October 2006.

[138] W. Sepp and G. Hirzinger. Real-time texture-based 3-d tracking. In *DAGM'03*, Magdeburg, Germany, 2003.

[139] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[140] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, July, October 1948.

[141] C. Shen, J. F. O'Brien, and J. R. Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. In *ACM Computer Graphics Proceedings, SIGGRAPH*, 2004.

[142] G. Slabaugh, W. Culbertson, T. Malzbender, M. Stevens, and R. Schafer. Methods for volumetric reconstruction of visual scenes. *International Journal of Computer Vision*, 2003.

[143] P. Smets. Belief functions and the transferable belief model. The Society for Imprecise Probability: Theories and Applications:

http://www.sipta.org/documentation/belief/main.html, March 2007.

[144] C. Stachniss and W. Burgard. Mapping and exploration with mobile robots using coverage maps. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003.

[145] C. Stachniss, G. Grisetti, and W. Burgard. Information gain-based exploration using Rao-Blackwellized particle filters. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[146] I. Stamos and P. Allen. 3-d model construction using range and image data. In *CVPR*, 2000.

[147] A. Stentz. The D* algorithm for real-time planning of optimal traverses. Technical Report CMU-RI-TR-94-37, The Robotics Institute, Carnegie Mellon University, Pittburgh, Pennsylvania 15213, September 1994.

[148] M. Stilman, J.-U. Schamburek, J. Kuffner, and T. Asfour. Manipulation planning among movable obstacles. In *Proceedings IEEE International Conference on Robotics and Automation*, Rome, Italy, April 2007.

[149] D. Stoyan and W. S. Kendall. *Stochastic Geometry and Its Applications*. J. Wiley, 1995.

[150] K. H. Strobl and G. Hirzinger. Optimal Hand-Eye Calibration. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4647–4653, Beijing, China, October 2006.

[151] K. H. Strobl, E. Wahl, W. Sepp, T. Bodenmueller, J.F. Seara, M. Suppa, and G. Hirzinger. The dlr hand-guided device: The laser-stripe profiler. In *Int. Conference on Robotics and Automation ICRA 2004*, New Orleans, U.S.A., 2004.

[152] W. Stuerzlinger. Imaging all visible surfaces. In *Proceedings of Graphics Interface*, 1999.

[153] V. A. Sujan and S. Dubowsky. Efficient information-based visual robotic mapping in unstructured environments. *International Journal of Robotics Research*, 24(4):275–293, April 2005.

[154] M. Suppa and G. Hirzinger. A novel system approach to multisensory data acquisition. In *The 8th Conference on Intelligent Autonomous Systems IAS-8*, Amsterdam, The Netherlands, 2004.

[155] M. Suppa and G. Hirzinger. Ein Hand-Auge-System zur multisensoriellen Rekonstruktion von 3d-Modellen in der Robotik. *at-Automatisierungstechnik*, 53(7):323–331, 2005.

[156] M. Suppa and G. Hirzinger. Multisensorielle Exploration von Roboterarbeitsräumen. In *Informationsfusion in der Mess- und Sensortechnik*. J. Beyerer (Hrsg.) and F. Puente León (Hrsg.) and K.-D. Sommer (Hrsg.), 2006.

[157] M. Suppa and G. Hirzinger. Multisensorielle Exploration von Roboterarbeitsräumen. *TM-Technisches Messen*, 74(3):139–146, March 2007.

[158] M. Suppa, S. Kielhoefer, J. Langwald, F. Hacker, K. H. Strobl, and G. Hirzinger. The 3D-Modeller: A multi-purpose vision platform. In *In Proceedings of International Conference on Robotics and Automation (ICRA)*, Rome, Italy, April 2007.

[159] M. Suppa, P. Wang, K. Gupta, and G. Hirzinger. C-space exploration using sensor noisy models. In *Int. Conference on Robotics and Automation ICRA 2004*, pages 4777–4782, New Orleans, U.S.A., 2004.

[160] K. Tarabanis, P. Allen, and R. Tsai. A survey of sensor planning in computer vision. *IEEE Transactions on Robotics and Automation*, 11(1), February 1995.

[161] K. Tarabanis, R. Y. Tsai, and P. K. Allen. The mvp sensor planning system for robotic vision tasks. *IEEE Transactions on Robotics and Automation*, 11(1), February 1995.

[162] K. Tarabanis, R. Y. Tsai, and A. Kaul. Computing occlusion-free viewpoints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(3):279–287, March 1996. Model-based approach.

[163] G. H. Tarbox and S. N. Gottschlich. Ivis: An integrated volumetric inspection system. *Computer Vision and Image Understanding*, 61(3):430–444, May 1995.

[164] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. Intelligent Robotics and Autonomous Agents series. MIT Press, 2005. http://www.probabilistic-robotics.org.

[165] S. Thrun, C. Martin, Y. Liu, D. Haehnel, R. Emery Montemerlo, C. Deepayan, and W. Burgard. A real-time expectation maximization algorithm for acquiring multi-planar maps of indoor environments with mobile robots. *IEEE Transactions on Robotics and Automation*, 20(3):433–442, 2003.

[166] S. Thrun, S. Thayer, W. Whittaker, C. Baker, W. Burgard, D. Ferguson, D. Haehnel, M. Montemerlo, A. Morris, Z. Omohundro,

C. Reverte, and W. Whittaker. Autonomous exploration and mapping of abandoned mines. *IEEE Robotics and Automation Magazine*, 11(4), 2005.

[167] A. P. Tirumalai, B. G. Schunck, and R. C. Jain. Evidential reasoning for building environment maps. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(1):10–20, January 1995.

[168] G. Turk and M. Levoy. Zippered polygon meshes from range images. In *SIGGRAPH*, 1994.

[169] G. Turk and J. F. O'Brien. Shape transformations using variational implicit surfaces. In *ACM Computer Graphics Proceedings, SIGGRAPH*, 1999.

[170] G. van den Bergen. A fast and robust gjk implementation for collision detection of convex objects. In *Journal of Graphics Tools*, 1999.

[171] G. van den Bergen. *Collision Detection in Interaction 3D Environments*. Morgan Kaufmann, November 2003.

[172] P.-P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Viewpoint selection using viewpoint entropy. In T.Ertl, B. Girod, G.Greiner, H. Niemann, and H.-P. Seidel (Eds.), editors, *Vision, Modeling, and Visualization*, pages 273–280, 2001.

[173] P.-P. Vázquez, M. Feixas, M. Sbert, and A. Llobet. Viewpoint entropy: A new tool for obtaining good views of molecules. In *Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization D. Ebert, P. Brunet, I. Navazo (Editors)*, 2002.

[174] E. Wahl, U. Hillenbrand, and G. Hirzinger. Surflet-pair-relation histograms: a statistical 3d-shape representation for rapid classification. In *Proceedings International Conference on 3-D Digital Imaging and Modeling – 3DIM*, pages 474–481. IEEE Computer Society Press, 2003.

[175] E. L. Waltz and J. Llinas. *Multisensor Data Fusion*. Artech House, 1990.

[176] P. Wang. View planning via maximal c-space entropy reduction. Master's thesis, School of Engineering Science, Simon Fraser University, April 2003.

[177] P. Wang and K. Gupta. Computing c-space entropy for view planning based on beam sensor model. In *International Conference on Intelligent Robots and Systems (IROS)*, Lausanne, Switzerland, 2002.

[178] P. Wang and K. Gupta. View planning via maximal c-space entropy reduction. In *Fifth International Workshop on Algorithmic Foundations of Robotics (WAFR)*, Nice, France, 2002.

[179] P. Wang and K. Gupta. Computing c-space entropy for view planning based on generic sensor model. In *IEEE International Conference on Robotics and Automation (ICRA)*, Taipei, Taiwan, 2003.

[180] P. Wang and K. Gupta. A configuration space view of planning. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, Bejing, China, 2006.

[181] P. Wang, R. Krishnamurti, and K. Gupta. Metric view planning with traveling cost and visibility range. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, Rome, Italy, April 2007.

[182] P. Wang, R. Krishnamurti, and K. Gupta. View planning with combined viewing and traveling costs. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, Rome, Italy, April 2007.

[183] S. Wenhardt, B. Deutsch, and J. Denzler. An information theoretic approach for next best view planning in 3-d reconstruction. In *IEEE International Conference on Computer Vision and Pattern Recognition (ICPR)*, Hongkong, China, 2006.

[184] F. Wewel, F. Scholten, and K. Gwinner. High Resolution Stereo Camera (HRSC) - multispectral 3d-data acquisition and photogrammetric data processing. *Canadian Journal of Remote Sensing*, 26(5):466–474, 2000.

[185] P. Whaite and F. P. Ferrie. Uncertain views. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR*, pages 3–9, 1992.

[186] P. Whaite and F. P. Ferrie. Autonomous exploration: Driven by uncertainty. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):193–205, 1997.

[187] P. Wunsch, G. Koegel, K. Arbter, and G. Hirzinger. Kalibrierung eines nichtlinearen, binokularen Hand-Auge Systems. *Technical Report IB-515-96-27*, 1996.

[188] Y. Yu and K. Gupta. An information theoretic approach to viewpoint planning for motion planning of eye-in-hand systems. In *31st International Symposium on Industrial Robotics (ISR) 2000*, Montreal, Canada, 2000.

[189] Y. Yu and K. Gupta. Sensor-based probabilistic roadmaps: Experiments with an eye-in-hand system. In *Journal of Advanced Robotics*, 2000.

[190] Y. Yu and K. Gupta. An information theoretic approach to view planning with kinematic and geometric constraints. In *International Conference on Robotics and Automation (ICRA)*, Seoul, Korea, 2001.

[191] Y. Yu and K. Gupta. On eye-sensor based path planning for robots with non-trivial geometry/kinematics. In *International Conference on Robotics and Automation (ICRA)*, Seoul, Korea, 2001.

[192] Y. Yu and K. Gupta. C-space entropy: A measure for view planning and exploration for general robot-sensor systems in unknown environments. *International Journal of Robotics Research*, 23(12):1197–1223, December 2004.

[193] S. Zelinka and M. Garland. Permission grids: Practical, error-bounded simplification. *ACM Transactions on Graphics*, 21(2):207–229, April 2002.

[194] H. Zhang, D. Minocha, T. Hudson, and K. E. Hoff III. Visibility culling using hierachical occlusion maps. In *SIGGRAPH*, 1997.

Tabellarischer Lebenslauf

Persönliche Daten

Anschrift	Michael Suppa Adenauerstr. 10 82178 Puchheim
Geboren	13.01.1976 in Hannover
Staatsangehörigkeit	deutsch
Familienstand	ledig

Schulbildung

1982–1986	Grundschule Hemmingen
1986–1995	Carl-Friedrich-Gauß Schule, Hemmingen
1995	Abitur

Wissenschaftlicher Werdegang

1995–2000	Studium der Elektrotechnik an der Universität Hannover
2000	Abschluß: Dipl.-Ing.
Seit 11/2000	wissenschaftlicher Mitarbeiter am Institut für Robotik und Mechatronik des Deutschen Zentrums für Luft- und Raumfahrt (DLR) in Oberpfaffenhofen
07/2003–11/2003	Gastwissenschaftler an der Simon Fraser University, Vancouver (Prof. Kamal Gupta)
Seit 10/2004	Koordinator der 3-D Modellierungs- und Bildverarbeitungsgruppe am Institut für Robotik und Mechatronik des DLR

München, den 22. Februar 2008 Michael Suppa