SENSOR POSITIONING IN DISTRIBUTED SENSOR NETWORKS

BY

NICOLAJ KIRCHHOF

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND INFORMATION TECHNOLOGY OF
TU DORTMUND UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

DOKTOR DER INGENIEURWISSENSCHAFTEN

(DOCTOR OF ENGINEERING)

September 2016

DEPARTMENT OF ELECTRICAL ENGINEERING AND INFORMATION TECHNOLOGY

SUPERVISORS:
Prof. Dr. Uwe Schwiegelshohn
Prof. Dr. Jörg Blankenbach

LOCATION:
Dortmund

ORAL EXAM:
12. September 2016

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHM

ACRONYMS

ABBREVIATIONS

ASP	Additional sensor poses
AWPN	Additional workspace positions
BIP	Binary Integer Programming
BLE	Bluetooth low energy
CMA-ES	Covariance Matrix Adaptation Evolution Strategy
CR	Conference room
FOV	Field of view
GDOP	Geometric Dilution of Precision
GNSS	Global Navigation Satellite System
ILP	Integer Linear Programming
LF	Large flat
LP	Linear Programming
MIP	Mixed Integer Programming
#SP	Number of sensor poses
#SSP	Number of selected sensor poses
#WPN	Number of workspace positions
OF	Office floor
RCPD	Radial Convex Polygon Decomposition
SC	Pairwise sensor combination
SF	Small flat
SP	Sensor pose
SPN	Sensor position
SPP	Sensor placement problem
ThILo	Thermal Infrared Localization
VF	Visibility function
VFOV	Visible field of view
WCAR	Worst case approximation ratio
WPN	Workspace position

MODELS

BSPQM	Best Sensor Pairwise Quality Model
CMQM	Continuous Maximum Quality Model
GCS	Greedy Combined Selection
GSSS	Greedy Single Sensor Selection
MSPQM	Minimum Sensor Pairwise Quality Model
RCPD-B	RCPD combined with BSPQM
RCPD-M	RCPD combined with MSPQM
SCO	Sensor Combination Optimization
STCM	Simple Two-Coverage Model

MATHEMATICAL NOTATIONS

A	$= \{a_1, \dots\}$	Set
$\mathfrak{A}$	$= \{A_1, \dots\}$	Familiy of sets (set of subsets)
$\boldsymbol{A}$		Matrix
A^{name}	$\boldsymbol{A}^{\text{name}}$	Set, matrix with additional naming
$\mathfrak{A}_i$	A_i	i^{th} set of a family of sets, element of a set
A_{ij}		$(i,j)^{\text{th}}$ element of a matrix
$\boldsymbol{a}$		Vector
a_i		i^{th} element of a vector
$\|\boldsymbol{a}\|$	$= \|\boldsymbol{a}\|_2$	Euclidean vector norm
$\mathbf{0}, \mathbf{1}$		Zero, one matrix. Zeros, ones in all elements
$\boldsymbol{a}_i^{\text{name}}$	$= (a_1, \dots)$	A named and indexed ordered tuple (n-tuple)
$[\boldsymbol{a}_i^{\text{name}}]_j$		The j^{th} element of a named and indexed n-tuple
$\|A\|$		The cardinality of a set (number of elements)

GEOMETRIC PRIMITIVES

$\boldsymbol{p}$	$= \left(\begin{smallmatrix} p_x \\ p_y \end{smallmatrix}\right)$	A point
$\boldsymbol{p}_i$	$= \left(\begin{smallmatrix} x_i \\ y_i \end{smallmatrix}\right)$	An indexed point (short form)
$\boldsymbol{e}$	$= (\boldsymbol{p}_a, \boldsymbol{p}_b)$	An edge
$\boldsymbol{\rho}$	$= (\boldsymbol{p}, \beta)$	A ray
$\boldsymbol{r}$	$= (\boldsymbol{e}_1, \dots)$	A ring
P	$= \{\boldsymbol{r}_1, \dots\}$	A polygon
$\mathfrak{P}$	$= \{P_1, \dots\}$	A multi polygon

GEOMETRIC OPERATIONS

$\|\boldsymbol{e}\|$	$= \|\mathbf{p}_i - \mathbf{p}_j\|$	The length of an edge
$\|a, b\|$		The shortest distance between two primitives
$\|P\|$		The area of a polygon
$\beta_{i,j}$	$\Leftrightarrow \beta(\mathbf{p}_i, \mathbf{p}_j)$	The bearing of the ray from $\mathbf{p}_i$ through $\mathbf{p}_j$
γ_{hij}	$\Leftrightarrow \gamma(\mathbf{p}_h, \mathbf{p}_i, \mathbf{p}_j)$	The inner bearing at $\mathbf{p}_i$ between $(\mathbf{p}_i, \mathbf{p}_h)$, $(\mathbf{p}_i, \mathbf{p}_j)$
$a \sqcap b$		The intersection of two geometric primitives
$a \sqcup b$		The union of two geometric primitives
$\overline{\wedge}(\mathbf{p}, P)$		The inner angle of vertex $\mathbf{p}$ of polygon P
$\overline{\overline{\wedge}}(P)$	$= \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$	All convex vertices of polygon P

SYMBOLS

$\boldsymbol{w}$	$= \binom{w_x}{w_y}$	Workspace position
$\mathbf{s}$	$= \begin{pmatrix} s_x \\ s_y \\ s_\varphi \end{pmatrix}$	Sensor pose
C	$= \{\mathbf{s}_h, \mathbf{s}_i\}$	Sensor combination
W	$= \{\boldsymbol{w}_j \mid \forall j\}$	All workspace positions
S	$= \{\mathbf{s}_i \mid \forall i\}$	All sensor poses
$\mathfrak{C}$	$= \{\{\mathbf{s}_h, \mathbf{s}_i\} \mid \forall h \forall i\}$	All sensor combinations
$\mathfrak{r}$		Radius of a sensing range
$\mathfrak{s}$		Angular range of a spike
$\mathfrak{q}$		Minimum positioning quality
$\mathfrak{a}$		Angular resolution
$\mathfrak{g}$		Grid cell length
ψ		Angular width of a FOV
φ		Angular offset to VFOV
Ψ_i	$\coloneqq \Psi(\mathbf{s}_i)$	VFOV polygon of a sensor $\mathbf{s}_i$
$a(h, i)$	$\coloneqq a(\mathbf{s}_h, \mathbf{s}_i)$	Avoidable overlap function
$q(h, i, j)$	$\coloneqq q(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}_j)$	Continuous SP-SP-WPN quality function
$q^b(c, j)$	$\coloneqq q^b(C_c, \boldsymbol{w}_j)$	Binary SC-WPN quality function
$\nu(i, j)$	$\coloneqq \nu(\mathbf{s}_i, \boldsymbol{w}_j)$	SP-WPN-Visibility function
$\boldsymbol{x}^s$		Decision vector used for SP
$\boldsymbol{x}^z$		Decision vector used for SC

1 INTRODUCTION

Throughout the ages, the simple question "where am I?" has led humankind to a comprehensive number of innovations. Among them are lighthouses, the sextant for celestial navigation and recently the Global Navigation Satellite System (GNSS). In all ages, the tools for self-positioning have been valuable goods. Nowadays, the positioning remains a problem indoors because GNSS-Signals are highly attenuated by walls (Kjærgaard et al., 2010).

Indoor positioning systems can be built upon nearly every sensor type. Prototypes have been developed or proposed in literature based on laser distance measurements, camera recordings, thermal infrared radiation and various other sensor input. Recently, the specification of the Bluetooth low energy standard gave indoor positioning implementations a new push. The technology allows to create cheap indoor positioning systems with an accuracy of approximately 5 m (Zhao, Xiao, Markham, Trigoni, & Ren, 2014).

The applications that may profit from indoor positioning are found in various domains, from navigation support for blind passengers at Airports (Iozzio, 2014) to automatic tool control in big production lines (Ubisense, 2015).

Taken into account that many applications are known and there exist extensive research on the techniques to localize people indoors, few systems have been developed to a market-ready state. Examples of companies that have built their business model solely on their positioning systems or positioning algorithms are Ubisense, indoo.rs and Nanotron. A vast comparison of indoor positioning publications and developed systems can be found in (Mautz, 2012). In addition, competitions like the EvAAL (Potort, Barsocchi, & Girolami, 2015) and competitive studies as exemplary provided in (Lymberopoulos et al., 2015) regularly provide information on the performance of newly developed positioning systems.

The only common aspect of most indoor positioning systems is their dependence on a network of active or passive sensors. The sensor signals from multiple sources make it possible to localize an object via triangulation, trilateration, multilateration or fingerprinting. The properties of the environment, the number of sensors that observe an object as well as their placement influence the accuracy of the compared positions. Keeping in mind the business aspect of the positioning scenario, a cost effective indoor positioning solution needs the fewest number of the cheapest sensors that can provide sufficient positioning results for its application.

Because the price of the sensors is usually determined by the choice of technology, a valuable method to reduce the overall system cost is to reduce the number of applied sensors. This is possible with the sensor placement optimizations proposed in this thesis.

1.1 MOTIVATION

Today, the positioning of sensor nodes is usually carried out using the system applicant's best guess. She decides how many sensors are placed and where to place them. Her decisions are based on her prior knowledge of the system parameters and the indoor geometry of the positioning area.

In the late 2000s, the Thermal Infrared Localization (ThILo) system (Kemper, 2010) was developed at the Institut für Roboterforschung. It is based on sensing the direction to heat sources from different positions, a classic triangulation scenario. The purpose of the system is to locate humans in indoor environments and its placement was always done in the described way, by simply choosing poses in the environments and placing the sensors.

Placement optimizations were already carried out manually, as the sensors were usually put in the edges of the environment to cover as much space as possible. A typical experimentation setup of the ThILo system is shown in Figure 1. Each of the ThILo sensors contains a line array of eight thermal infrared sensitive pixels, whereas each pixel has a FOV of 6°. Therefore, the overall FOV of the sensor is 48° by 6°.

To localize people in an indoor environment, the sensors are placed in chest height (approximately 1.3 m) on the boundary of the environment. Here, two sensors are needed to cover a 90° corner and the whole environment, as shown in the example, can be covered with eight sensors. By exploiting the specific sensor dampening function, as described in (Hauschildt, Kemper, Kirchhof, Juretko, & Linde, 2010), the angle to a heat source can be estimated with an angular error of $\pm$ 1.5°.

With the described system setup, most ThILo experiments were carried out. Nevertheless, the question of how good the sensor configuration really is, had not yet been answered.

The research that led to the presented thesis was initially driven by the desire to have a metric that shows the quality of the ThILo placement. It quickly shifted in a more general direction, to provide means of placing sensors in the best possible way while taking the geometric properties of positioning techniques into account. In addition, task requirements like the definition of a minimum positioning quality were included. In this thesis, the used task requirements and sensor properties are deducted from the ThILo system.

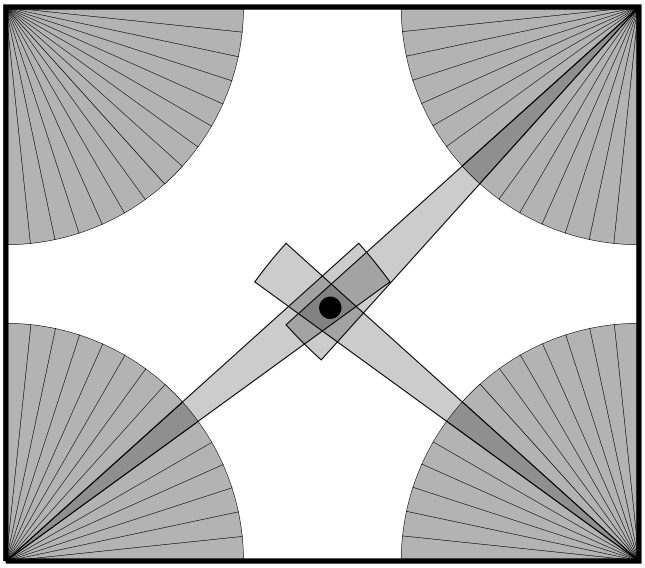


Figure 1: Default setup of the ThILo positioning system with four sensor pairs that are placed at the edges of a room. Each sensor pair contains two sensors with eight pixels each and a total FOV of 90°. The pixels are visualized as gray cones. In addition, a localized object is present in the middle of the room and three of the sensor pixels with a strong input signal are highlighted. Only the full detection range of pixels with a strong input signal is drawn to allow a better differentiation between the sensor pairs.

1.2 OUTLINE

The scope of this thesis contains methods to compute the sensor poses of a positioning system from a digitalized environment map, as shown in Figure 2. It is structured in the following way: in Chapter 2, the background of the used methods and notations is summarized and the problem setting is introduced. In Chapter 3, all calculations and algorithms that were applied or developed to state a sensor placement model are presented. Especially discretization and quality metrics are described in detail. Altogether, it shows how a digitalization of a real world environment can be transformed into an environment representation that serves as input for the proposed sensor positioning algorithms.

Chapter 4 describes how optimization models can be stated from sensor placement problems based on the environment representation. Multiple models are presented that provide solutions with different levels of exactness and complexity. Chapter 5 delineates approximations and heuristics for the problems, to provide methods able to find solutions with reasonable complexity. Two basic strategies are explored, approximation and problem partitioning, more specific: divide and conquer. Chapter 6 concludes the presented methods by introducing placement models based on the full geometric information

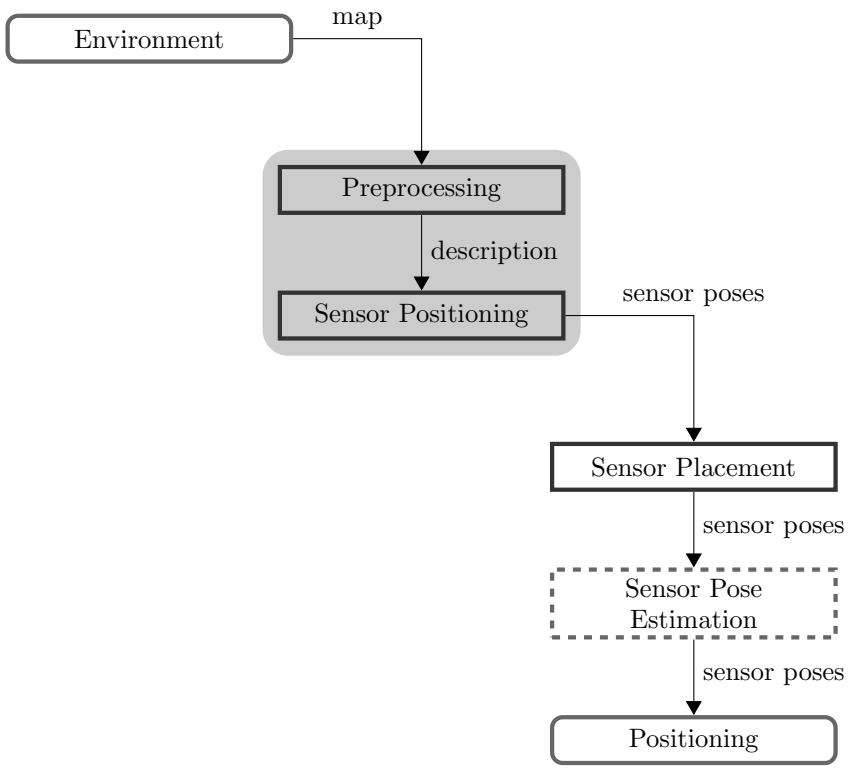


Figure 2: Overview of a positioning system setup. At first, the positioning environment is to be mapped to a digital 2d environment representation. This has to be preprocessed for the sensor placement algorithms to ensure that every object is tagged appropriately and a set of possible sensor poses and workspace positions along with other input parameters are available. Then, a sensor positioning algorithm computes the fewest sensors to be placed while ensuring that a predefined positioning accuracy can be met. The result is used to place real sensors in the environment. Optionally, their pose can be calibrated with additional sensor pose estimation algorithms like the one proposed by Kemper, Walter, and Linde (Kemper, Walter, & Linde, 2008). Finally, the sensor poses are used to parameterize a positioning engine that computes real world positions from the sensor data.

of the environment by making use of a continuous problem representation.

To evaluate the research on sensor positioning, Chapter 7 presents the experiments that were conducted to validate and compare the proposed methods. Finally, the research is summarized in Chapter 8 and possibilities for further research are proposed.

1.3 CONTRIBUTION

The proposed methods in this thesis are contributions to the field of sensor positioning and computational geometry. They contain discrete optimization models and heuristics to solve the problem of sen-

sor selection from pre-calculated sensor poses under task constraints. The models are developed with respect to the properties of real-world vision-based positioning systems. Therefore, they handle sensor restrictions in detection range (angular and distance) and geometrical distribution.

All models are based upon the premise that a predefined positioning environment is to be fully covered with the least possible number of sensors.

In detail, the contributions include:

- A novel discretization scheme that gives a stable geometric representation of the samples.
- A refined positioning quality metric optimized to solve sensor placement problems.
- Three discrete optimization models to solve sensor placement problems with different levels of accuracy.
- A simplified optimization model and three heuristics to approximately solve sensor placement problems along with the derivation of the WCAR for two of them.
- A search strategy to improve approximate solutions.
- A novel approach to decompose a 2d polygon with holes into convex parts allowing Steiner Points only on the boundary.
- Two fitness functions that provide means of rating the quality of arbitrary sensor configurations.
- An evaluation based on computer simulations of the proposed methods.

2 BACKGROUND

This chapter introduces the methods used throughout the dissertation. At first, a short introduction on sensor positioning is provided that shows the related and relevant work (Section 2.1). In the remaining of this chapter, basic notations and important theorems in the relevant research fields on computational geometry (Section 2.3), integer programming (Section 2.4) and positioning quality metrics (Section 2.5) are introduced. Finally, the chapter concludes with the presentation of an environment model that includes all relevant parameters for the sensor positioning problems (Section 2.6).

2.1 SENSOR POSITIONING

Sensor positioning is the art of finding the best configuration for one or more sensor nodes while considering the task they apt to fulfill. A famous example is the GPS satellite placement on orbits in a manner that every point on earth has to be covered by at least four satellites for it to work. To solve this task, Walker developed different configurations in the early 1970s (Walker, 1971) that provide valid solutions. These have been improved by optimization approaches over time using the increasing computational power and new mathematical methods (Lang & Adams, 1998).

At the same time as Walker developed satellite constellations, the American mathematician Victor L. Klee posed the problem of how to determine the minimum number of "guards" sufficient to cover every wall in an art gallery (Honsberger, 1976). To solve this problem, the art gallery is assumed a simple two-dimensional (2d) polygon and a guard is a point within the polygon that can see in every direction until its view is blocked by a wall. In essence, the question is: what would be the lowest number of star shaped polygons[1] the simple polygon can be decomposed to?

Following Klee's proposal, many theoretical problems of similar kind arose. Most of them either constrain the art gallery to a special polygon type (rectangular/star shaped), or apply restrictions on the guards. For example, a guard can only be placed at the vertices (edges) of the art gallery polygon, or its view is limited in distance or angular range (Franklin, 1989).

Art gallery problems were mostly of theoretical relevance until distributed sensor networks came into play in the early 1980s (Labora-

1 Star shaped polygons are polygons that contain at least one point from which every other point in the polygon is visible. See Section 2.3.4 for further details.

tory, 1986). With the technology available at that time, a sensor node was approximately the size of a shoebox and weighted kilograms. Positioning and target tracking were the focus of these networks and their development was strongly pushed by the Defense Advanced Research Projects Agency (DARPA) for military applications. Over time, the sensor nodes shrank to only a few centimeters and below (Chong & Kumar, 2003).

In recent years, a trend in sensor-positioning research is the development of cheap Wireless Sensor Networks with nodes that include visual sensors. The nodes are used for surveillance applications and have the ability to adjust their field of view (FOV). A sensor network of such nodes is supposed to work autonomous in a target area, which leads to objectives like energy-efficient communication and distributed organization of the node orientations (Cai, Lou, Li, & Li, 2009). In this context, another objective has been introduced, the term "connectivity" (Kouakou, Yamamoto, & Yasumoto, 2010). It means that the placed sensors have to be no further apart than their wireless range to ensure that they are connected to each other (Mini, Udgata, & Sabat, 2012). The research has focused on calculating the number of sensors that have to be randomly deployed in a positioning area to provide either full (Song, Ding, Kamal, Farrell, & Roy-chowdhury, 2011) or so-called "barrier"-coverage (Zhang, Tang, & Zhang, 2009). Latter is the coverage along one direction in the positioning area, which prohibits an undetected crossing.

Distributed sensor networks that are used for indoor positioning applications were introduced in the late 1990s and early 2000s (Hightower & Borriello, 2001). They used wireless technologies like Bluetooth (Beutel, Kasten, & Ringwald, 2003) or radio frequency transmission in combination with Ultrasound (Balakrishnan, Supervisor, & Smith, 2005). Over time, an increasing number of sensor and communication techniques have been exploited for indoor positioning applications by Mautz (Mautz, 2012).

Indoor positioning systems can be divided in passive and active systems. Active systems are mostly based on radio frequency technologies like Wireless LAN, Radio Frequency Identification or Ultra-Wideband (Deak, Curran, & Condell, 2012). In recent years, the Bluetooth low energy (BLE) technology gave active positioning systems a strong push (Contreras, Castro, & Torre, 2014). Furthermore, the omnipresence of smartphones and their ability to utilize BLE signals lead to a new ecosystem of hardware manufacturers[2] that provide cheap BLE "beacons" and software application developers that provide the positioning applications[3].

In contrast to the active systems, passive system utilize sensors that allow a device-free positioning of humans based on signals like

2 Examples are: Kontakt.io and Estimote

3 Examples are: Indoo.rs and Contagt

sound, visible light, thermal infrared or electric field interference (Kivimäki, Vuorela, Peltola, & Vanhala, 2014). The advantage of these systems is that the localization target, typically a human, serves as the signal source for the positioning system and no additional hardware is necessary. Nevertheless, signals like visible light and thermal infrared require a line-of-sight between the sensor and the sensed human, which has to be provided by the placement of the sensors.

The development of distributed sensor systems for positioning applications gave the art gallery algorithms a practical use. Especially in indoor scenarios, the representation of a floor plan by a simple 2d polygon is evident. In addition, the variety of sensor systems and their technical details demands for specialized sensor placement problems that represent them. Thus, sensor placement has been well studied in literature. Mittal and Davis (Mittal & Davis, 2007) and Tarabanis, Allen, and Tsai (Tarabanis et al., 1995) made extensive surveys of the research that was conducted in this field. They categorize the sensor-positioning approaches into three categories, based on the available information about the environment.

The first category contains problems where no prior information is available and the placement strategies are based on increasing the environment information successively using the available information for the next placement step. The second category contains problems with only partial information about the objects and sensors of the environment, while in the third category complete information about the environment is a priori available.

The context of this thesis is set in the third category. It encloses sensor positioning approaches based on exact information of the geometric properties of the environment and the sensor properties. Two objectives dominate the research in this category. The first one is to restrict the number of sensors either directly or by using a cost measure along with a maximum cost to compute the maximum coverage possible for the environment. The second one is based on the prerequisite of a full workspace coverage and tries to decrease the number of sensors directly or via a cost metric. Hörster and Lienhart (Hörster & Lienhart, 2006b) compare these metrics for an indoor environment with cameras. In addition, Kim (Kim, 2007) combines both to form a multi-objective optimization solution.

Both objectives are usually combined with additional constraints, introduced by the sensor or environment model of the system. Commonly used constraints are:

CONNECTIVITY (of the sensor nodes) is used to position autonomous wireless sensor networks where the communication range differs from the sensing range (Han, Cao, Lloyd, & Shen, 2008).

VISIBILITY (of the sensor nodes) restricts the sensor ability to detect objects, which includes e.g. a limited FOV and a limited sensing distance. Both limitations are common especially in visual

sensor networks (Erdem & Sclaroff, 2004). Further, the ability to detect objects is another example for visibility constraints. It can be modeled by the probability of object occlusions (Mittal & Davis, 2004).

k-COVERAGE (of the environment) implies that every point in the environment is to be covered by k sensors (Kumar, Lai, & Balogh, 2008).

PRIORITIES (of regions in the environment) used in coverage maximization problems to define areas that are highly frequented or of special interest (Hörster & Lienhart, 2006b). These are given a higher probability to be covered. They can either be defined by the system engineer (Conci & Lizzi, 2009) or automatically created, e.g. by extracting points of interest from the floor plan and using a path-finding algorithm to calculate ways that have a higher possibility of being used (Nam & Hong, 2012).

For positioning systems, a simple coverage of the environment is usually insufficient to compute an object's position, especially if the positioning system is based on triangulation or trilateration. Given a 2d environment representation, these techniques require at least measurements from two different sensor locations to calculate a target position with triangulation. To calculate an unambiguous target position using trilateration three different sensor positions are necessary. Thus, the models developed in this field usually imply k-coverage constraints (Kumar et al., 2008).

In addition, the application of the sensor positioning system might introduce quality constraints, e.g. a maximum tolerable positioning error. Such requirements directly influence the sensor positioning task, as they must be handled by a task satisfying distribution of the sensors. In this specific research area of sensor positioning with task constraints Isler and Bajksy (Isler & Bajksy, 2006) presented a selection problem, in which a subset of omnidirectional but range-restricted sensors has to be chosen to minimize the triangulation uncertainty at predefined positions. They used approximation algorithms, which keep a small intersection area of the 2d uncertainty polygons while reducing the overall size of active sensors. In contrast, Liu, Zhang, and Ma (Liu et al., 2011) use a probabilistic detection model to analyze the positioning quality of a given camera placement. Their model implies that cameras have a limited FOV and a detection probability based on the object distance. Furthermore, a position in the environment is sufficiently covered, if $k \geqslant 2$ cameras cover it and their mean estimated error is below a system-specific threshold. Finally, Tekdas and Isler (Tekdas & Isler, 2010) use the uncertainty of the localization based on the geometric distribution of the sensors, namely the GDOP to optimize the placement of sensors that are used for target position-

ing. However, they only consider omnidirectional sensors and a very small input size.

The models and heuristics developed within this thesis are also determined to enable an indoor positioning with a desired quality through sensor positioning. The constraints that are embedded into the proposed models are those for typical visual sensor systems.

2.2 THE SENSOR PLACEMENT PROBLEM

After prior work to solve sensor placement problems has been introduced, the sensor placement problem (SPP), which is the basis for the research described in this thesis, can be formally defined.

Definition 1. *The sensor placement problem that is handled within this thesis is to place the lowest possible number of sensors, restricted in sensing range and angular range, on the walls and suitable furniture of an environment to cover the accessible area of the environment with a sufficient localization quality.*

The main goal of the definition is to place as few sensors as possible. In addition, the "accessible area" refers to every part of an environment that is accessible for the targets that are localized by the positioning system. Within this thesis, the targets are people. Therefore, large furniture like tables or cupboards define places at which the sufficient localization quality is not needed.

The restrictions of the sensors are common for visual sensor systems. They need to be considered by ensuring that furniture, which blocks the FOV of a sensor, is modeled accordingly. To serve the specific sensor configuration of the ThILo system and to restrict the problem size, only the walls and suitable furniture are considered for sensor placement. The sensor placement restriction also serves the practical consideration that sensors need a support to be placed.

An additional restriction concerning real world sensor positioning problems is the transformation of the problem to a 2d domain. Within this thesis, a 2d domain is used for all models that are presented to solve defined SPP. This restriction is used to limit the size of the input problems but it is not compulsory for the presented algorithms to work. On the contrary, the proposed optimization models that use a discrete domain can easily be extended to handle a 3d representation of the SPP, e.g. by using approaches for visibility calculation from (Mittal & Davis, 2004).

2.3 MATHEMATICAL NOTATIONS

Most of the presented work is based on set operations and methods of computational geometry in 2d. This section provides the foundation for specific mathematical interpretations and definitions used in this

thesis. Especially the notation of polygon intersections and visibility calculations, along with the notations of the geometric primitives will be needed to follow the sensor-placement calculations.

2.3.1 *Sets and Tuples*

A basic set is an unordered collection of items and usually given in set builder notation (Rosen, 2011) as

$$X = \{x \mid x \in \mathbb{N}\},$$

whereas $\mathbb{N}$ is the set of all natural numbers. More complex conditions are combined using the binary notation of "and" ($\wedge$) and "or" ($\vee$). To give an example, the following equation states that $\mathfrak{X}$ is a family of sets of all combinations of natural numbers, whereas one of the numbers in each combination has to be an element of a predefined set Z as

$$\mathfrak{X} = \{\{x, y\} \mid (x \in \mathbb{N}) \wedge (y \in \mathbb{N}) \wedge (\exists z[(z \in \{x, y\}) \wedge (z \in Z)])\}$$

The term $\exists z[(z \in \{x, y\}) \wedge (z \in Z)]$ reads as "there exists a z such that z is an element of the set $\{x, y\}$ and z is an element of Z".

The common notations for intersection $\cap$, union $\cup$, removal $\setminus$ and inclusion (subsets) $\subset, \subseteq$ are used. In addition,

$$\bigcup \mathfrak{Z} \coloneqq Z_1 \cup Z_2 \cup \ldots, \quad \forall Z_i \in \mathfrak{Z}$$

is used as a short notation for the union of all subsets of a family of sets. If the subsets themselves have no further subsets, the union will result in a set of all unique elements of all subsets of $\mathfrak{Z}$. For example, given a family of sets

$$\mathfrak{Z} = \{\{a, b\}, \{b, c\}, \{c, d\}\},$$
$$\bigcup \mathfrak{Z} = \{a, b, c, d\}.$$

This operation is also called "flattening".

In contrast to sets, tuples have a defined order. Given a tuple

$$\mathbf{r} = (a, \cdots, z),$$

common set notations are used to define tuple manipulations. For example, the notation

$$\mathbf{r} \setminus ([\mathbf{r}]_h, \cdots, [\mathbf{r}]_k)$$

or

$$\mathbf{r} \setminus (h, \cdots, k)$$

is used to describe the removal of a subsection of the tuple. The order of a tuple is specified using the notation

$$(x_1 < \cdots < x_n),$$

which implies that an element x_i in the tuple has a value less than an element x_j if $i < j$.

2.3.2 *Geometric Primitives*

The simplest primitive in the 2d environment is a point

$$\mathbf{p}_i = \begin{pmatrix} p_{x_i} \\ p_{y_i} \end{pmatrix} \Leftrightarrow \begin{pmatrix} x_i \\ y_i \end{pmatrix}.$$

It is represented by a two-element vector. An edge

$$\boldsymbol{e} = (\mathbf{p}_a, \mathbf{p}_b)$$

is a tuple of two points. To refer to the first or second point of the edge the notations

$$[\boldsymbol{e}]_1 \coloneqq \mathbf{p}_a, \qquad [\boldsymbol{e}]_2 \coloneqq \mathbf{p}_b$$

are used.

Whereas an edge is defined by its two points, a ray

$$\boldsymbol{\rho} = (\mathbf{p}, \beta). \tag{1}$$

is only bounded by one point $\mathbf{p}$ and defined as an infinite line that runs in a direction β, its bearing.

An extension of an edge is a ring, which is a sequence of connected, non-intersecting edges

$$\begin{aligned} \mathbf{r} &= (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n) \\ &= ((\mathbf{p}_1, \mathbf{p}_2), (\mathbf{p}_2, \mathbf{p}_3) \ldots, (\mathbf{p}_{n-1}, \mathbf{p}_n), (\mathbf{p}_n, \mathbf{p}_1)). \end{aligned}$$

The orientation of a ring, which is the succession of points, is either clockwise or counter-clockwise around the enclosed interior.

One or more non-intersecting rings can be combined to form a polygon

$$P = \{\mathbf{r}_1, \ldots, \mathbf{r}_n\}.$$

An example of a polygon is drawn in Figure 3. Only the first of the polygon rings—the boundary $\mathbf{r}_1$—has a clockwise orientation. All other rings define holes in counter clockwise orientation and must be within $\mathbf{r}_1$. The "active area" of a polygon is defined as every point within the boundary and not within a hole.

Finally, a compound of polygons is called a multi polygon and defined as a set of polygons

$$\mathfrak{P} \coloneqq \{P_1, \ldots, P_n\}.$$

There are no additional constraints on the polygons in a multi polygon, which means they do not need to be spatially separated and are allowed to intersect.

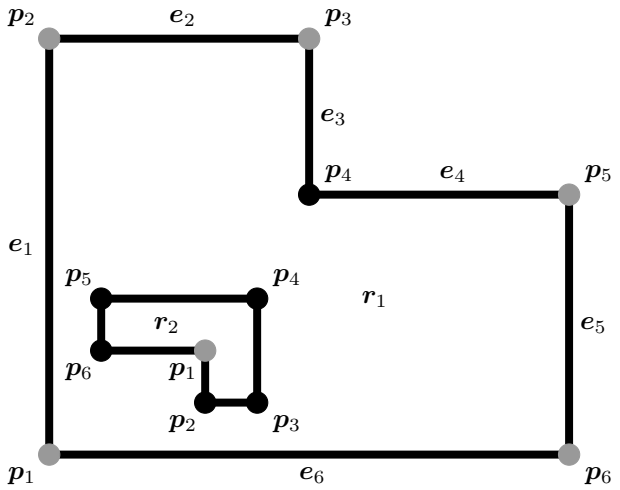


Figure 3: The Figure shows a polygon represented by an outer ring $\mathbf{r}_1$ and an inner ring (hole) $\mathbf{r}_2$. The vertices are highlighted and colored black if they represent a convex vertex; otherwise, they are colored in gray. The convex vertices are $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_5, \mathbf{p}_6$ on the outer ring and $\mathbf{p}_1$ on the hole. Furthermore, the vertex numbering is shown for both rings and the edge numbering for the boundary.

2.3.3 *Geometric Operations*

A point of a ring is usually called vertex or node. The vertices of a polygon can be classified as "reflex" or "convex" based on the angle between the edges they connect. For two edges

$$\begin{aligned} \boldsymbol{e}_{i-1} &= (\mathbf{p}_{i-1}, \mathbf{p}_i) \\ \boldsymbol{e}_i &= (\mathbf{p}_i, \mathbf{p}_{i+1}) \end{aligned}$$

the angle of vertex $\mathbf{p}_i$ is measured between the edges $\boldsymbol{e}_{i-1}$ and $\boldsymbol{e}_i$ on the side facing the active area. Two edges with an angle of $[0°, 180°]$ define a convex vertex and two edges with an angle of $(180°, 360°]$ define a reflex vertex. Within this thesis

$$\overline{\wedge}(\mathbf{p}, P)$$

is defined to be a function that returns the inner angle of the given vertex $\mathbf{p}$, which must be an element of the given polygon P. In addition,

$$\overline{\overline{\wedge}}(P) \coloneqq \{\mathbf{p} \mid (\mathbf{p} \in P) \wedge (\overline{\wedge}(\mathbf{p}, P) \in [0°, 180°])\}$$

denotes the geometrical operation, to extract the convex vertices of a polygon P. In Figure 3 the convex vertices are highlighted.

Geometric operations on primitives will be denoted using set alike notations. For example, an intersection of an edge $\boldsymbol{e}_i$ with a polygon P that results in a set of intersection points

$$I = \{\mathbf{p}_1, \cdots, \mathbf{p}_n\}$$

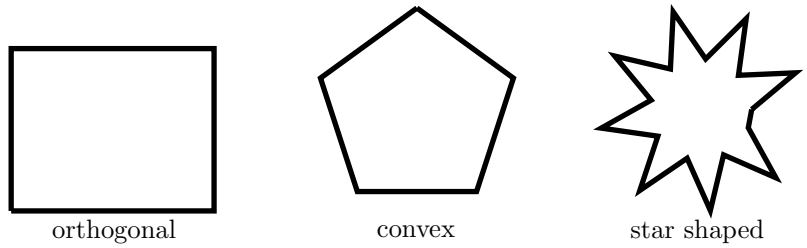


Figure 4: Examples of polygons labeled with their type.

will be denoted as

$$I = \boldsymbol{e}_i \sqcap P.$$

In essence, it is the intersection of $\boldsymbol{e}_i$ with every $\boldsymbol{e} \in P$. In contrast, the intersection of two polygons P_x, P_y is a multi polygon

$$\mathfrak{Z} \coloneqq P_x \sqcap P_y.$$

Thus, the outcome of the graphical intersection operator will depend on the context and explained when used.

To denote the union of two geometric primitives, the notation

$$a \sqcup b$$

is used. The symbol

$$a \sqsubset b$$

is used to denote that a primitive a is within a primitive b. It is usually used to indicate that a point is within a polygon. To denote that the primitive can also be on the boundary, the symbol $\sqsubseteq$ is used.

The Euclidean norm is used to define the length of an edge as

$$\|\boldsymbol{e}\| \coloneqq \|\boldsymbol{p}_b - \boldsymbol{p}_a\|_2.$$

It also denotes the size of the active area of a polygon as $\|P\|$.

The norm of two geometric primitives is used as an expression of a function that returns the shortest distance between them. Thus, let $\mathbf{r}$ and $\mathbf{s}$ be two geometric primitives and $f^d(x, y)$ be a function that returns the shortest distance between two geometric primitives then

$$\|\mathbf{r}, \mathbf{s}\| \coloneqq f^d(\mathbf{r}, \mathbf{s}).$$

2.3.4 *Types of Polygons*

Polygons can be classified based on their shape. In Figure 4, three commonly used polygon shapes convex, star shaped and orthogonal are displayed.

The orthogonal polygon is only allowed to have orthogonal or 90° connections of adjacent edges. In contrast, the difference between the convex and star shaped polygon is based on their visibility properties.

In a convex polygon, every point is "visible" from every other point, whereas visibility is defined using a straight line that connects both points. If this line is completely inside the polygon, both points are visible to each other. This also implies that a convex polygon does not have any reflex vertices (Ghosh, 2007).

A star-shaped polygon has reflex vertices but it must contain a point or region from which every other point in the polygon is visible. In Figure 4, this region is in the center of the pictured star-shaped polygon.

All of them are sub classes to a forth polygon type, the simple polygon that was already shown in Figure 3. A simple polygon has no restrictions on visibility and may contain holes. In addition, an orthogonal polygon can be a convex polygon, if it is a rectangle, a convex polygon is always also a star shaped one and an orthogonal might be one, if it does not have two adjacent reflex vertices (Goodman & O'Rourke, 2004).

2.3.5 *Algorithms on 2d Polygons*

From the polygon definition, it is only a small step to define a geometrical description of a 2d floor plan. For this purpose, the positioning area is typically modeled as a simple polygon with holes, which represent objects that are opaque for the sensor nodes like furniture or interior walls.

The detection area of a sensor—further called the visible field of view (VFOV)—with a limited range can be modeled as a star-shaped polygon, usually even as a convex polygon[4]. Putting both, the floor plan and the sensing range representation together, the single sensor coverage problem is similar to the problem of polygon coverage, which is well known in algorithmic geometry. It is proven to be NP-hard (Goodman & O'Rourke, 2004), if the polygon to be covered is simple and may contain holes.

In literature, the problem is tackled by using predefined placement patterns for which a worst-case placement can be calculated (Bai, Kumar, Xuan, Yun, & Lai, 2006), by solving binary integer optimization models (Osais, St-Hilaire, & Yu, 2009), by heuristically solving non-linear optimization models like simulated annealing (Mittal & Davis, 2004) or heuristics like the greedy algorithm in combination with custom-placement metrics (Hörster & Lienhart, 2006a).

A problem with the direct transformation of a sensor placement problem into a polygon coverage problem is object visibility. For a visual sensor system, the detection range cannot be calculated by sim-

4 The calculation of VFOV polygons is explained in Section 3.2.

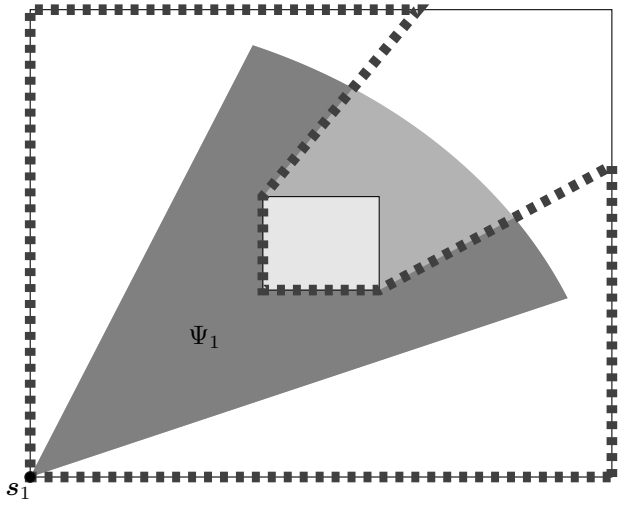


Figure 5: The sensor located at the lower left corner ($\mathbf{s}_1$) has the gray FOV. Because it overlaps the rectangular obstacle in the center, the part of it that is located behind the obstacle is not visible. To account for these situations, the visible area has to be calculated from $\mathbf{s}_1$, shown with the dotted line. The intersection of the FOV and the visible area gives the VFOV (Ψ_1), the part of the environment visible from $\mathbf{s}_1$.

ply placing the sensors visible field of view (VFOV) inside the floor plan and calculating the intersection of both polygons. In Figure 5, it is shown that the intersection polygon also contains a section behind an obstacle that was placed in the floor plan, which is not visible from the sensors point of view. To solve this problem, additional visibility calculations have to be performed.

Computing the visibility from any point in the polygon can be done using a ray and performing an angular sweep with it. In the process, the visibility to the polygon vertices of the simple polygon is computed following a clockwise or counter-clockwise direction. Based on the vertex visibilities, the edges visible to the sensor position form the VFOV. The complexity of this computation is $\mathcal{O}(n \log n)$, whereas n is the number of polygon vertices (Ghosh, 2007).

Beside the visibility calculations and covering problems, two other important complexity statements will be used within this thesis. The first one are polygon operations, which are in essence the intersection, union and difference of two polygons. The complexity of these operations is $\mathcal{O}(nm)$, whereas n and m are the number of edges of the polygons to be combined (Greiner & Hormann, 1998). The second one is polygon decomposition, which usually has its applications in computer graphics e.g. for polygon model simplifications (Lien & Amato, 2008). Depending on the input polygon properties and the desired decomposition, the complexity of optimal decomposition algorithms range from triangulation problems that can be solved in $\mathcal{O}(n)$, for simple polygons without holes (Chazelle, 1991) and $\mathcal{O}(n \log n)$ for simple polygons with holes (Hertel & Mehlhorn, 1985), to the decom-

position into convex polygons, which can be computed in $\mathcal{O}(n^3)$ for simple polygons without holes but is NP-hard, if the simple polygon has holes (Chazelle & Palios, 1994).

2.3.6 *Algorithm Notations*

Because of the excessive use of algorithms within this thesis, common algorithm notations are introduced. Within algorithms, the notation $\leftarrow$ is used for an assignment such as

$$x \leftarrow 1$$

and $\coloneqq$ is used to define a function as

$$f(x) \coloneqq x + 1$$

or the state of a variable as

$$L \coloneqq (a, \cdots, z).$$

Latter is mostly used in loops to allow complex operations on ordered tuples such as:

```
1 repeat
2 |  x ≔ (x_1, x_2, ..., x_n)
3 |  x ← (x_2, ..., x_n, x_1)
4 until forever
```

In this example the first element of the tuple is shifted to the last position in the first execution of Line 2 and 3. In the next execution of Line 3, the former x_1 is now x_n by the definition in Line 2.

2.4 INTEGER PROGRAMMING

Beside the usage of algorithmic geometry, a part of this dissertation will focus on modeling the sensor placement problem (SPP) using integer programming models. These are solved using a state of the art branch and cut based solver. Since Integer Linear Programming (ILP) methods are commonly known, their solving techniques are not repeated but the interested reader is referred to one of the reference works like Watkins (Watkins, 1990). In this section, only some key facts about ILP are summarized.

In general, solving ILPs is a NP-hard task. It is usually based on solving a Linear Programming (LP) relaxation of the ILP problem and approaching the global optimal discrete solution via branch and cut techniques. If the LP solution contains only integer values, an optimal solution for the ILP is found. Otherwise, the LP solution is used to apply additional constraints that split the problem at non-integer values of the solution.

A special case of ILP is Binary Integer Programming (BIP) that uses only integer variables constrained to the values $\{0, 1\}$ (Bradley, Hax, & Magnanti, 1977). Beside the ILP and BIP, the third kind of optimization problem used in this thesis is Mixed Integer Programming (MIP), which contains a mix of discrete and continuous optimization variables. Solving BIP and MIP models are both NP-hard tasks (Karp, 1972).

It has to be noted that throughout the thesis the term "model" will be used as a short form for a "mathematical model" as described in (Sarker & Newton, 2007). Usually the model will be further specified based on its properties e.g. as an BIP model or MIP model. To give an example, a BIP model is stated using the notation

$$\text{minimize} \quad \sum_{\forall i} x_i, \tag{2}$$

$$\text{subject to} \quad \sum_{\forall i} i|x_i| \geqslant 4, \tag{3}$$

is used whereas the variables $x_i \in \{0, 1\}$. The optimization criterion is shown in Equation 2, and the constraint in Equation 3.

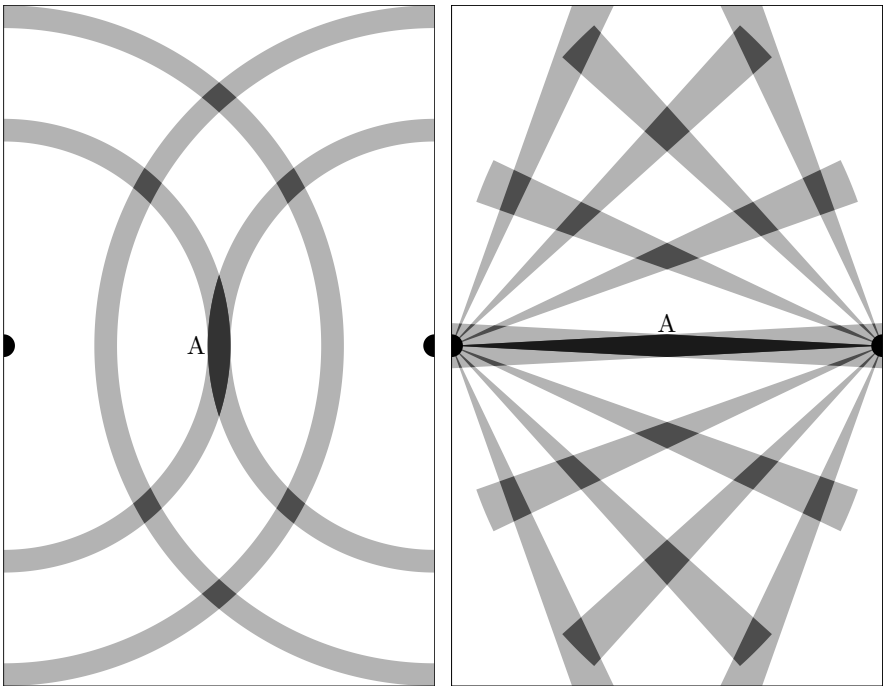
2.5 QUALITY METRICS

The influence of sensor placement on positioning uncertainty is introduced with a simple example presented in Figure 6. It shows the relation between the pose uncertainty at different workspace positions (WPNs) in relation to the sensor placement.

In a rectangular environment, two sensors are placed opposite of each other and their uncertainty polygons for targets in different directions and at different distances are sketched. It can be seen that on a straight line between the two sensors the positioning uncertainty is high due to the sensor geometry. Therefore, if the objects to be localized are located in this region, the positioning performance will suffer.

To calculate the uncertainty only introduced by the geometric properties of the sensor placement, Kelly (Kelly, 2003) proposed to use the Jacobian determinant of the triangulation or trilateration equations as the Geometric Dilution of Precision (GDOP) (Swanson, 1978). Commonly, the GDOP is extensively used in context of GNSS positioning (Zhu, 1992).

Kelly's proposed 2d GDOP leads to an uncertainty measure that is defined for every point in the workspace with respect to two sensors that observe it. Since the work is based on the ThILo system, which uses bearing measures to calculate the position of humans in indoor environments, the GDOP derivation for triangulation based systems is used. Here, the GDOP depends on the distance and the inverse of the inner bearing angle of two bearing measures from different sensors at an observed point.



(a) Trilateration (b) Triangulation

Figure 6: An example of the uncertainty at different positions in a positioning environment, where two sensor are placed opposite of each other at the walls of a rectangular room. For the trilateration example, the uncertainty of two distance measurements of each sensor are plotted as gray half circles. For the trilateration example the gray uncertainty cones of seven angular measurements are plotted. The dark gray intersection regions serve as examples for the positioning uncertainty of an object that is located in the center of the region. The larger dark area in the middle—depicted as "A"—indicates an area with a low position accuracy. The smaller gray areas in the top and bottom half indicate areas with a high position accuracy (Kelly, 2003).

The work presented in this thesis makes use of the GDOP as an indicator of the positioning accuracy in relation to the geometric distribution of the sensors and their measurement error. Thus, the GDOP provides the basis for the metric on which the positioning quality is measured. To parameterize this metric, a static definition of the measurement error in combination with the desired application of the positioning system is used. For the ThILo system, the desired application is activity monitoring in context of ambient assisted living.

2.6 ENVIRONMENT DESCRIPTION

To conclude the background chapter, this section presents the models that were defined to restrict and structure the input extracted from the SPP. It shows which data has to be provided to successfully execute the placement algorithms that were developed.

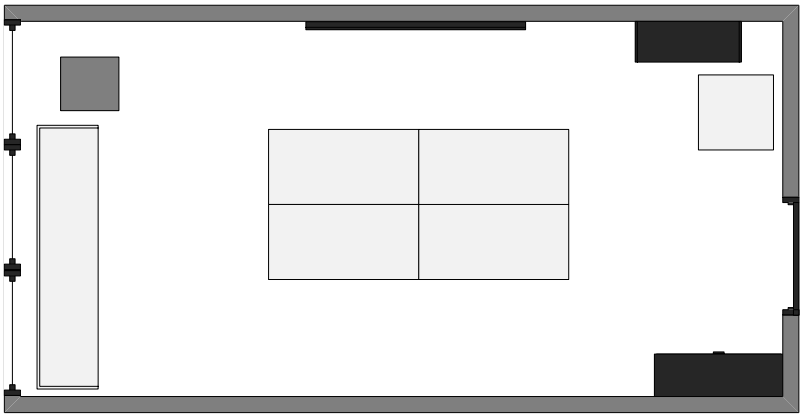


Figure 7: A floor plan that can be processed by the sensor placement algorithms. The floor plan shows a conference room, in which the light gray areas are occupied regions where tables and a cupboard are placed. The black areas are obstacles that have to be considered in the visibility analysis. The dark gray areas are the surrounding walls and objects that can be mounted with sensors.

In general, the representation of a real environment—in this case an indoor environment—can either be in form of a 2d or a 3d model. For both it has to be assured that the representation contains all important features needed for further processing (Fallah, Apostolopoulos, Bekris, & Folmer, 2013). To map real world environments, there are different automatic map approaches suggested in literature, which are not handled in further detail in the dissertation. The interested reader is referred to (Thrun, 2002). In this thesis, an up to date and valid floor plan is assumed to be provided by the user.

In addition to the environment, the sensor has to be modeled. According to its properties, this can be done through an algorithmic or mathematical description of the sensors behavior. Alternatively, a simple static geometrical representation can be used, e.g. a polygon of a circle or a ball with the dimension of the sensor range. Both can be evaluated up to an arbitrary exactness, but for the geometrical representations usually a fixed precision is used to allow robust geometric operations (Schirra & Schirra, 2000).

2.6.1 *Spatial Model*

The spatial model describes the environment where the positioning takes place. In geometric terms, the environment is represented by a simple polygon (Goodman & O'Rourke, 2004) that may contain holes. In practice, such a floor plan can be created using standard software for vector graphics.

To distinguish the polygons of the floor plan, annotations have to be provided for any polygon and any hole. These provide additional in-

formation for the placement algorithms. In the visual representation of the environment, these objects are color coded, as shown Figure 7.

Furthermore, the spatial model has to meet some prerequisites to be valid:

1. It has to be surrounded by a closed wall.
2. The interior may not contain fully separated rooms. These rooms will be neglected, as they can be computed separately.
3. All objects in the interior have to be annotated with one of the following tags:

   OCCUPIED Polygons of objects or areas within the environment where positioning accuracy is not needed but that is opaque when placed in the FOV of a sensor.

   OBSTACLE Polygons of objects that occludes the region behind it, when it is placed in the FOV of a sensor. Obstacles can also occur as part of walls as can be seen on the left and right side of Figure 7. These are objects like doors or windows, where sensors cannot be placed.

   MOUNTABLE A freestanding object that allows the placement of sensors on its exterior and occludes the space behind it, when in the FOV of a sensor.

   WALL Polygon of the exterior of the environment.

It is assured that the placement algorithms work correctly, if their input meets the defined requirements.

Since every object is a polygon, every object class is represented by a multi-polygon. These multi-polygons will be referred to as:

$\mathfrak{P}^{\text{obs}}$	The multi polygon of all obstacles
$\mathfrak{P}^{\text{mnt}}$	The multi polygon of all mountables
$\mathfrak{P}^{\text{wall}}$	The multi polygon of all walls
$\mathfrak{P}$	The multi polygon of all space model objects

$\mathfrak{P}$ includes all objects that are placed within the environment, thus it can be calculated by building the union of all multi-polygons

$$\mathfrak{P} = \mathfrak{P}^{\text{occ}} \cup \mathfrak{P}^{\text{obs}} \cup \mathfrak{P}^{\text{mnt}}. \tag{4}$$

2.6.2 *Sensor Model*

In contrast to the spatial model, the sensor model is based on common properties of visual sensors. It is shown in Figure 8. The properties a sensor are defined as:

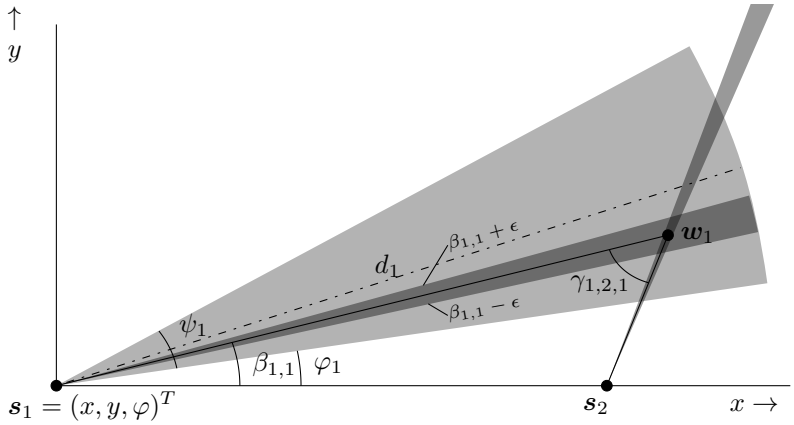


Figure 8: The model of a sensor, which is located at the coordinates (x, y) and is rotated with the angle φ in respect to the x-axis. Its FOV is defined by the parameter $\psi_1 = 20°$ and its distance by the parameter $d_1 = 9.5m$. In addition, a second sensor s_2 and a WPN w_1 is shown along with the bearing angle $\beta_{1,1}$ between s_1 and w_1. The dark gray cones that are plotted over the bearing from s_1 and s_2 to w_1 indicate the respective angular error range $\mathfrak{e}$ for an angular error of $\epsilon = 1.5°$ for both sensors. Finally, $\gamma_{1,2,1}$ is the inner bearing angle of s_1 and s_2 at w_1.

$\mathfrak{r}$ the sensing range of the sensor. It is modeled as a strict representation how far an object can be from the sensor until it is no longer detectable.

ψ the angular range, which defines how wide the FOV is.

$\mathbf{s}$ the pose of the sensor that is composed of the s_x and s_y coordinate and the orientation s_φ as

$$\mathbf{s} = \begin{pmatrix} s_x \\ s_y \\ s_\varphi \end{pmatrix}. \tag{5}$$

Restricted by the sensing distance $\mathfrak{r}$ and angular range ψ, the sensing range can geometrically be described by a convex polygon.

The measurement error of a sensor is modeled as an angular offset to the real target angle. Let β be the measured angle to a target and ϵ the angular error of a sensor, the angular range that holds the real angle to the target can be defined as

$$\mathfrak{e} \coloneqq [\beta - \epsilon, \beta + \epsilon]. \tag{6}$$

If the real angle to the target is given, as shown in Figure 8, the definition states the range of the measured angle.

3 PREPROCESSING

This chapter introduces the preprocessing steps to prepare indoor environments for the application of the developed sensor placement models. In addition, it introduces some newly developed and adapted discretization procedures for geometric models.

The content of this chapter is based on the three preprocessing steps that are necessary to model a sensor placement problem. First, the digital indoor map is used to calculate, a set of sensor poses (SPs) and a set of workspace positions (WPNs) as discrete variables (Section 3.1). Then, the visibilities between these variables are calculated (Section 3.2). These are used to calculate pairwise sensor combinations (SCs) that satisfy basic quality constraints (sections 3.3 and 3.4). Finally, detailed positioning quality metrics are introduced (Section 3.5).

3.1 DISCRETIZATION

The first preprocessing step on a spatial environment model is to calculate a set of SPs and WPNs. The SPs represent possible locations where sensors can be placed and the WPNs are points in the positioning environment where the positioning quality is measured. Both are essential variables, which are needed to form discrete sensor placement models. They are sampled from the spatial environment model in a user-defined quantity, which naturally has an impact on the quality of discrete models and their solution. The better the discrete output represents the continuous map, the closer an optimal discrete solution is to an optimal continuous one.

In general, sampling can be accomplished by means of randomly drawing from a probability distribution, by using a specific pattern (sampling function) or by using a heuristic (Lohr, 2009). The advantage of the first approach is that priorities can be considered by the choice and the parameters of the probability distribution. On the downside, the drawn variables are likely to change in subsequent model generations. Therefore, subsequent outcomes of this approach are difficult to compare.

In contrast, making use of a specific pattern or a heuristic, the discretization process can be executed in a controlled manner. A good example of this is the grid based sampling approach, where the sampled positions form a grid with uniform cell length. The advantage of this sampling strategy is that the density of samples is the same in all parts of the area. Using this sampling strategy will result in points

that have the same distance to their four neighboring points as shown in Figure 9. The only parameter to define a grid based sampling is the grid cell length (g) and a grid position that serves as anchor.

The disadvantage of using the grid based sampling method is the number of samples that depends on the cell size and increases quadratically if the cell size is halved. In addition, the positions of the workspace points change with the chosen grid cell length, regardless of an adjustment of the anchor. This behavior is not optimal considering the samples are supposed to be the basis for quality function evaluations. Hence, two models with a different number of samples state completely different optimization problems, whose outcomes are difficult to compare. Due to the ILPs strong increase in complexity in relation to the input size, the quadratic increase for stable discretizations that divide the cell size by two is not well suited for time-boxed evaluations.

3.1.1 *Sampling of Workspace Positions*

The sampling of WPN has the goal to provide stable discretizations of the positioning area with the scene. Here, "stable" refers to the discrete positions sampled with different quantities. The algorithm is based on quadtrees (Finkel & Bentley, 1974) an approach, where a 2d environment is subdivided to an arbitrary number of cells by splitting it recursively into four equally sized quadrants. It was adapted to provide stable discretizations based on an initial cell length g, which defines the minimum number of samples necessary to give a coarse representation of the positioning area.

The adapted grid sampling strategy is shown in Algorithm 1. It is based on a sampling function $\omega(g)$ (Line 1) that samples all points, which are:

- multiple of the grid size $(\frac{p_x}{g} \in \mathbb{N}) \wedge (\frac{p_y}{g} \in \mathbb{N})$,
- not within any object $(\mathbf{p} \not\sqsubset \mathfrak{P})$,
- inside the boundary $(\mathbf{p} \sqsubset \mathfrak{P}^{\text{wall}})$.

Initially, all points are sampled with the given grid cell length (Line 2). Then, the algorithm iteratively decreases the grid cell size (Line 6) and samples new WPNs that are not already in W (Line 7). By testing if the remaining number of additional WPNs m is less or equal to the number of sampled WPNs in W^{new} (Line 3) before adding them to W (Line 4), the algorithm ensures that W never holds more WPNs than requested. In contrast, if $m < |W^{\text{new}}|$ the loop is quit and the remaining grid positions are selected in an iterative manner (Line 11 to 15). To accomplish this, an ordered tuple is created that holds an ordered tuple containing the selected WPNs of W^{new} sorted in as-

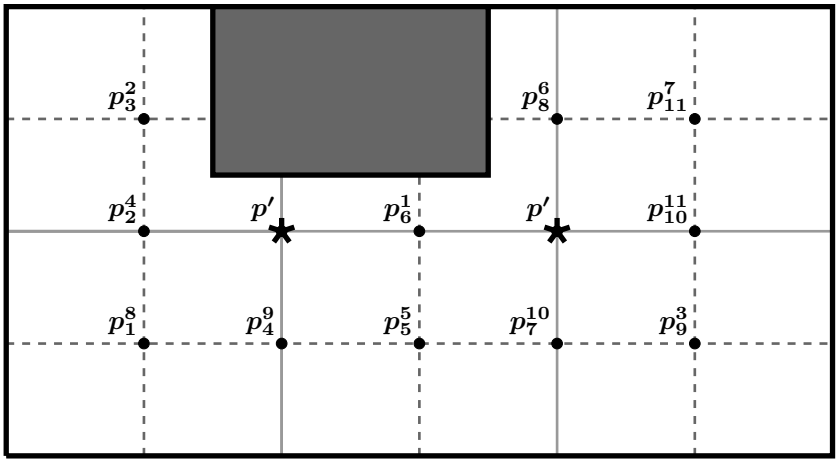


Figure 9: Adapted grid based sampling shown for a simple floor plan of a room. The black star marks labeled as $\mathbf{p'}$ are the initial grid positions, which are sampled on the initial grid depicted as gray lines. The black dots are the WPNs sampled in the first iteration of the grid refinement. Their subscript indicates the order of the WPNs in the initial L_1 tuple and their superscript indicates the number in which the additional points are selected.

cending order (Line 9), which is defined to be the order of the WPNs x-coordinate first and then the y-coordinate as

$$\mathbf{p}_i < \mathbf{p}_{i+1} \Leftrightarrow (x_i < x_{i+1}) \vee ((x_i = x_{i+1}) \wedge (y_i < y_{i+1})).$$

Because of their grid structure, the points are sorted in increasing y-order for the grid x-coordinates, which are in turn added in increasing order. A sample of the ordering is shown in Figure 9.

For the initial ordered tuple, the index of the midpoint is calculated in (Line 13) and the point, which is somewhere near the center of the grid, is added to W (Line 15). The separation of the initial ordered tuple at the midpoint (Line 14) and the reassignment of the two newly created tuples spatially distributes the next midpoint selections. In subsequent runs WPNs are added to W that more likely to have a high distance from each other. An example is given in Figure 9, here at first the center point chosen, then the upper left point, the lower right point and so forth.

Altogether, the placement scheme preserves the advantages of the grid-covering scheme while providing a defined number of points to be placed inside the environment. In addition, with every additional point, the grid is refined but the WPNs are stable. Thus, for two discretizations with a different number of WPNs

$$|W_x| < |W_y| \Rightarrow W_x \subset W_y.$$

Data: A number m of additional positions
An initial cell size g
An environment defined by its walls $\mathfrak{P}^{wall}$ and objects $\mathfrak{P}$
Result: A set W of sampled WPNs

$\omega(g) \coloneqq \{\mathbf{p} \mid (\frac{p_x}{g} \in \mathbb{N}) \wedge (\frac{p_y}{g} \in \mathbb{N}) \wedge (\mathbf{p} \not\sqsubset \mathfrak{P}) \wedge (\mathbf{p} \sqsubset \mathfrak{P}^{wall})\}$
$W \leftarrow \omega(g), \quad W^{new} \leftarrow \emptyset$
while $m \geqslant |W^{new}|$ **do**
  $W \leftarrow W \cup W^{new}$
  $m \leftarrow m - |W^{new}|$
  $g \leftarrow \frac{g}{2}$
  $W^{new} \leftarrow \{\mathbf{p} \mid (\mathbf{p} \in \omega(g)) \wedge (\mathbf{p} \notin W)\}$
end
$L \leftarrow ((\mathbf{p}_1 < \ldots < \mathbf{p}_n)), \quad \forall \mathbf{p} \in W^{new}$
$W^{new} \leftarrow \emptyset$
while $|W^{new}| \leqslant m$ **do**
  $L \coloneqq (L_1, L_2, ..., L_x), \quad L_1 \coloneqq (\mathbf{p}_a, \ldots, \mathbf{p}_k, \ldots, \mathbf{p}_z)$
  $k \leftarrow \left\lceil \frac{|L_1|}{2} \right\rceil$
  $L \leftarrow (L_2, ..., L_x, (\mathbf{p}_a, \ldots, \mathbf{p}_{k-1}), (\mathbf{p}_{k+1}, \ldots, \mathbf{p}_z))$
  $W^{new} \leftarrow W^{new} \cup \mathbf{p}_k$
end
$W \leftarrow W \cup W^{new}$

Algorithm 1: Adapted grid sampling strategy.

3.1.2 *Sampling of Sensor Poses*

In contrast to the sampling of WPNs, the sampling of SPs is restricted to the boundaries of the walls and mountable objects as explained in Section 2.2. Nevertheless, none of the SPP algorithms requires the SPs to be solely placed on the boundary. Still, the computational advantage of using sensor poses restrained to the boundary is the significant reduction in the number of possible SPs in contrast to a sampling everywhere in the environment.

Similar to the sampling of the WPNs, the sampling of the SPs is conducted in an iterative process that provides a user-defined increase in the number of sampled SPs. Because not only positions but also poses are sampled, an additional parameter—namely the angular resolution a—is needed. It is used to determine the angular distance of sensor poses at a sampled sensor position and thus influences the number of poses created.

The sampling of SPs is based on an initial set of SPs. It contains the most promising spots for SPs, the corners. Especially the convex corners have the furthest distance from the polygon center, which increases the probability of a large environment coverage if sensors are placed at these points. Furthermore, the selection of the corners

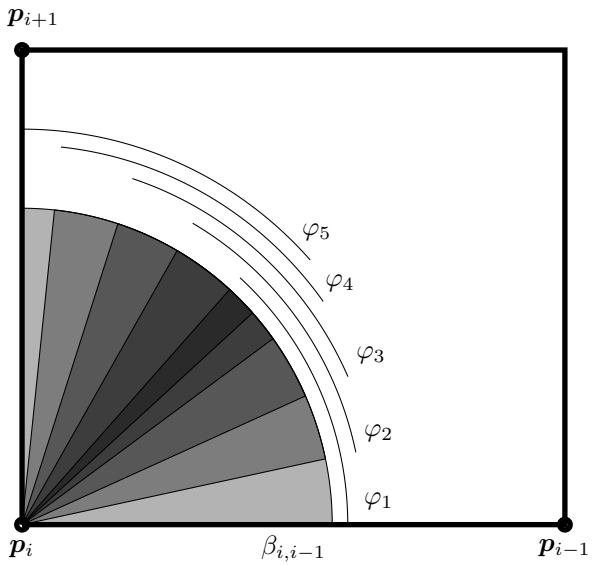


Figure 10: An example of sampled sensor poses for one vertex ($\mathbf{p}_i$) of an environment polygon in the lower left corner. The SPs with the offsets φ_1, φ_2, φ_3 and φ_4 are sampled in clockwise direction at an angle of $\beta_{i,i-1} = 0°$, $\beta_{i,i-1} + \alpha = 12°, \ldots$. The FOV of the sensor poses is 48°. The next sampled FOV at φ_5 is modified to fit inside the environment because it would start at an offset of 48° and therefore supersedes the environment boundary. Thus, its angular offset is changed to 42°=90°-48°.

has the advantage that it provides a spatial distribution of the initial SPs.

For a given vertex $\mathbf{p}_i$, its predecessor $\mathbf{p}_{i-1}$ and its successor $\mathbf{p}_{i+1}$, the sampling of SPs is performed along the angle $\gamma_{i-1,i,i+1}$ between the two edges $\boldsymbol{e}_{i-1}$ and $\boldsymbol{e}_i$ that $\mathbf{p}_i$ connects. Let α be a predefined angular resolution and $\beta_{i,i-1}$ be the angle between point $\mathbf{p}_i$ and $\mathbf{p}_{i-1}$ then $s(\mathbf{p}_i)$ can be defined as the function to sample a set of SPs at a given vertex $\mathbf{p}_i$ as

$$f^{sp}(\mathbf{p}_i) := \left\{ \begin{pmatrix} \mathbf{p}_i \\ \beta_{i,i-1} + \varphi \end{pmatrix} \;\middle|\; \varphi \in \{0, \alpha, \ldots, \gamma_{i-1,i,i+1} - \psi\} \right\}. \quad (7)$$

Here, the first sensor FOV is aligned with the incoming edge and the last FOV is aligned with the outgoing edge as shown in Figure 10.

The sampling of SPs is presented in Algorithm 2. It initially samples all SPs on convex vertices of the walls or mountables (Line 1) and assigns them to a set S. For further sampling of additional SPs, all edges that are on the boundary or on mountables are assigned to a tuple $\mathbf{l}$. The sorting order, depicted by $\succeq$, refers to the length of the edges (Line 2). Thus,

$$\boldsymbol{e}_i \succeq \boldsymbol{e}_j := \|\boldsymbol{e}_i\| \geqslant \|\boldsymbol{e}_j\|. \quad (8)$$

In each run of the loop (lines 4 to 11) the longest edge from the sorted list is selected and its midpoint is calculated (Line 9). At the mid-

Data: A number n of additional poses
The walls $\mathfrak{P}^{\text{wall}}$ and mountables $\mathfrak{P}^{\text{mnt}}$ of an environment
Result: A set S of sampled SPs

1 $S \leftarrow \{f^{sp}(\mathbf{p}) \mid \mathbf{p} \in \overline{\overline{\lambda}}(\mathfrak{P}^{\text{wall}} \cup \mathfrak{P}^{\text{mnt}})\}$
2 $\mathbf{l} \leftarrow (\boldsymbol{e}_i \succeq \boldsymbol{e}_{i+1} \succeq \ldots \succeq \boldsymbol{e}_n), \quad \forall \boldsymbol{e}(\boldsymbol{e} \sqsubseteq (\mathfrak{P}^{\text{wall}} \cup \mathfrak{P}^{\text{mnt}}))$
3 $S^{\text{new}} \leftarrow \emptyset$
4 **while** $|S^{new}| < n$ **do**
5 $\quad n \leftarrow n - |S^{\text{new}}|$
6 $\quad S \leftarrow S \cup S^{\text{new}}$
7 $\quad \mathbf{l} \coloneqq (\boldsymbol{e}_i \succeq \boldsymbol{e}_{i+1} \succeq \ldots \succeq \boldsymbol{e}_n)$
8 $\quad \boldsymbol{e}_i \coloneqq (\mathbf{p}_i, \mathbf{p}_{i+1})$
9 $\quad \mathbf{p}_m \leftarrow \mathbf{p}_i + \frac{\mathbf{p}_{i+1} - \mathbf{p}_i}{2}$
10 $\quad S^{\text{new}} \leftarrow f^{sp}(\mathbf{p}_m)$
11 $\quad \mathbf{l} \leftarrow (\boldsymbol{e}_{i+1} \succeq \ldots \succeq (\mathbf{p}_i, \mathbf{p}_m) \succeq (\mathbf{p}_m, \mathbf{p}_{i+1}) \succeq \ldots \succeq \boldsymbol{e}_n)$
12 **end**
13 **if** $|S^{new}| > n$ **then**
14 $\quad S \leftarrow S \cup S^{\text{new}}_{1,\ldots,n}$
15 **end**

Algorithm 2: Sensor pose selection algorithm.

point additional SPs are sampled (Line 10). The midpoint is then regarded as an additional 'virtual' vertex that splits the edge in two parts, which are put back into the sorted list of edges (Line 11). If the number of additional SPs $|S^{\text{new}}|$ exceeds the remaining number of additional SPs, only the required number of SPs is selected from the last sampled sensor position (Line 14).

After the sampling, new SPs are to be checked with respect to their VFOV, the region of the environment seen from the SPs. If a VFOV does not contain any WPNs and hence would not add any value to an SPP model it is to be removed. This post-processing can be expressed using a validation function for sampled SPs as

$$\Gamma(\mathbf{s}_i) = \begin{cases} 1, & \text{if the VFOV polygon of } \mathbf{s}_i \text{ contains at least one WPN,} \\ 0, & \text{otherwise.} \end{cases}$$

The function is used to filter S as

$$S = S \setminus \{\mathbf{s} \mid (\mathbf{s} \in S) \wedge (\Gamma(s) = 1)\}. \tag{9}$$

3.2 VISIBILITY

The sampling of WPNs and SPs leads to a discrete representation of the environment. To express the relationships between them, the visibilities of the SPs in relation to the WPNs have to be calculated.

Ghosh (Ghosh, 2007) presents different algorithms that allow a visibility calculation in 2d environments. The presented algorithms differ in their applicable domain. For a simple polygon that may contain holes, a $\mathcal{O}(n \log n)$ plane sweep algorithm is given. A similar technique to the one stated by Erdem and Sclaroff (Erdem & Sclaroff, 2006).

In essence, a plane sweep uses a ray that originates from the point for which the visibility is calculated. The direction of the ray is swept in a circular "movement" once around the whole 360° angular range. During the sweep, the intersections with the polygon boundary are used to compute the boundary of visibility polygon.

An example of such a visibility polygon has been stated in Figure 5. In contrast to the FOV, which can be computed solely from the sensor parameters, the visibility polygon is solely based on the environment properties and a given position. To get a representation of the exact area that is within the sensor range and visible to the sensor, the intersection of the visibility polygon and FOV-polygon has to be computed. This results in the visible field of view (VFOV), which represents both the environmental and sensor influence on the visibility. The computational effort to perform such a polygon intersection is $\mathcal{O}(|E_1||E_2|)$ (Greiner & Hormann, 1998).

3.2.1 *Spikes*

Geometric calculations like polygon intersections and visibility polygon generations may result in polygons that contain spikes, especially if both calculations are combined. In general, a spike is a vertex that connects two edges, which run almost parallel and are very close to each other. If the vertex is on the outside of the adjacent edges, it is called outer spike, otherwise it is an inner spike. For the VFOV polygons, the outer spike definition is extended to a spiked region that may include more than one vertex. Thus, a spiked region is a part of the outer polygon ring of two or more edges that form an opening with an angular diameter smaller than the maximum spike range ς. An example of a polygon with two-spiked region is shown in Figure 11. Here, the left region contains two edges and the right region six edges.

Spiked regions usually occur in environments with obstacles or reflex edges during the calculation of sensor VFOVs. Thus, they represent areas where the sensor only sees small regions beside an obstacle or a reflex edge. In real world environments such visibilities a very unreliable because it is unclear, if the observed information of an object that is present in a spiked region is enough to recognize it. For example, for a camera positioning system the object may just be covered by a few pixels and might be unrecognizable by a feature detection algorithm.

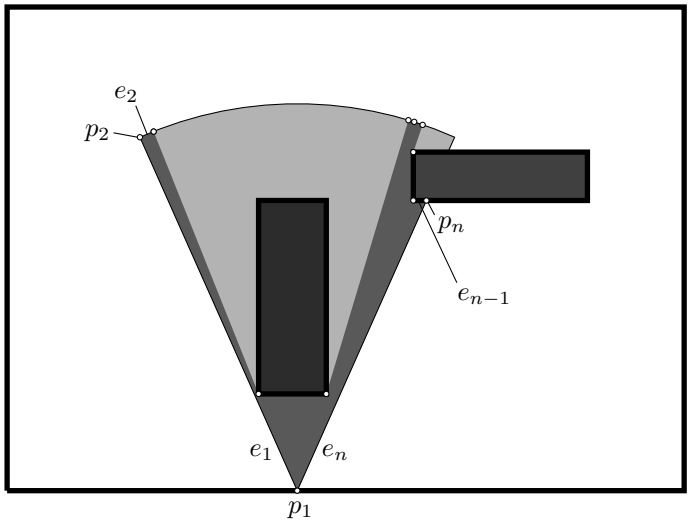


Figure 11: A sensor that is placed on the boundary and has an obstacle in front of it and two-spiked regions in its VFOV. The first spiked region on the left side only contains two vertices, whereas the spiked region on the right side contains six vertices because of the second obstacle that is also in the region of this spiked region. The first $\mathbf{p}_1$, second $\mathbf{p}_2$ and last $\mathbf{p}_n$ vertex of the VFOV and the first $\boldsymbol{e}_1$ and last $\boldsymbol{e}_n$ edge is marked.

Nevertheless, a WPN within a spiked region is computationally visible, which may lead to unreliable outcomes of SPP solutions. Thus, spiked regions have to be removed prior to any SP-WPN visibility calculation.

To remove them, an algorithm based on the geometric properties of the VFOV polygons is proposed. These polygons are always star shaped because every point has to be visible from the sensor position. Therefore, they do not contain holes and only contain a single ring that defines the polygon boundary.

The spike removal procedure works in two steps. First, all neighboring points that are too close to each other are merged without moving the outer points that define the FOV. This converts spiked regions to spikes that can be categorized in three classes: "mergeable", "left-oriented" and "right-oriented". They are presented in Figure 12. The mergeable spikes consists of two vertices close to each other with a vertex between them that spikes to the outside of the polygon. The spike can be removed by merging the two vertices on the outside of the spike. In contrast, the left and right-oriented spikes have one of the vertices close to the edge that is defined by the other two. These can be removed by merging the vertex onto the edge. Afterwards, the spiked vertex can be removed from the polygon.

The procedure consists of two parts. At first, the angular point merge (Algorithm 3) is responsible for merging points that are too close to each other. It works on the outer ring r of a VFOV polygon where it runs from the second vertex $\mathbf{p}_2$ and the last vertex $\mathbf{p}_n$ (see

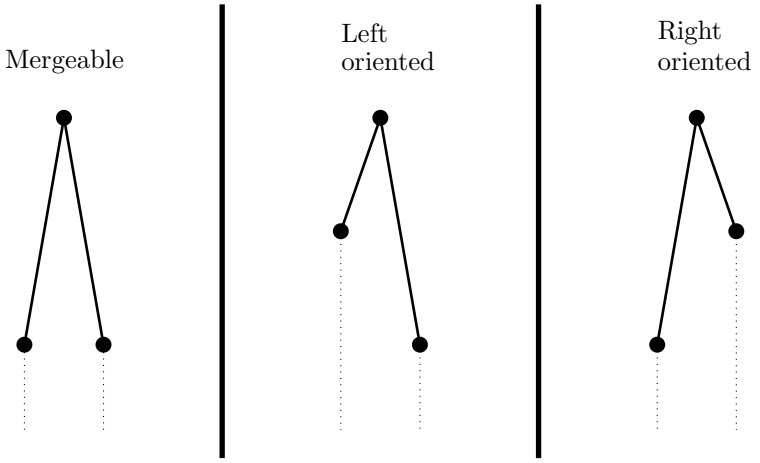


Figure 12: The three types of spikes that may exist in the VFOV polygon.

Figure 11). The algorithms main loop runs from the second to the n^{th} point of the ring (lines 3 to 14). Within the loop it is repeatedly tested if the edge between the current point $\mathbf{p}_j$ and a point further along the ring $\mathbf{p}_{j+k}$:

- exist, because $\mathbf{p}_{j+k}$ is a valid point of the ring (Line 5),
- does not intersect the environment (Line 6),
- has an angular length is less or equal to s (Line 7).

If the initial edge at point $\mathbf{p}_j$ was increased by at least one point (Line 8) a spike merge can be performed. Therefore, edge $\boldsymbol{e}_j = [\mathbf{r}]_j$ is replaced by a newly created edge between $(\mathbf{p}_j, \mathbf{p}_{j+k})$ (Line 11). In addition, all edges between $\mathbf{p}_{j+1}$ and $\mathbf{p}_{j+k}$ are removed from the ring.

The second part of the procedure (Algorithm 4) cleans possible spiked regions without affecting the FOV defining vertices $\mathbf{p}_2$ and $\mathbf{p}_n$ of the polygon. It takes three vertices starting with $\mathbf{p}_2$, $\mathbf{p}_3$ and $\mathbf{p}_4$ and checks them for the three types of spikes (Line 7, 12, 20). Instead of using the angular spike range directly, the algorithm calculates the maximum allowed distance at the spike start and endpoint (Line 5). If they match the criteria of any of the three types of spikes, the algorithm handles them by removing the middle vertex of a mergeable spike (Line 8-9) or shifting the middle vertex to the closest position from the first or last vertex for a left and right oriented spike (Line 15-16, 23-24). Latter is only executed if the new edge does not intersect with the environment, which is checked in line 14 and 22. Depending on the performed action, the next vertices are selected in Line 10, 17, 25 or 29.

Beside the distance criteria, both algorithms only merge and remove vertices, if the changes do not lead to a polygon edge, which intersects an obstacle, a mountable or a wall of the environment. Thus,

Data: A VFOV polygon boundary ring $\mathbf{r} = (\boldsymbol{e}_1, \cdots, \boldsymbol{e}_n)$ in which $\mathbf{p}_1$ is the SP.
An angular spike range $\mathfrak{s}$
Result: The polygon boundary ring $\mathbf{r}$ cleaned of all possible angular spikes.

$\mathfrak{P}^{\text{env}} \leftarrow \mathfrak{P}^{\text{obs}} \cup \mathfrak{P}^{\text{wall}} \cup \mathfrak{P}^{\text{mnt}}$
$j \leftarrow 2$
while $j \leqslant n$ **do**
  $k \leftarrow 2$
  while $(j + k \leqslant n)$
    $\wedge(\emptyset = \bigcup\{(\mathbf{p}_j, \mathbf{p}_{j+k}) \sqcap P \mid P \in \mathfrak{P}^{env}\})$
    $\wedge(\gamma(\mathbf{p}_j, \mathbf{p}_1, \mathbf{p}_{j+k}) \leqslant \mathfrak{s})$ **do**
    $k \leftarrow k + 1$
  end
  if $(k > 2)$ **then**
    $[\mathbf{r}]_j \leftarrow (\mathbf{p}_j, \mathbf{p}_{j+k})$
    $\mathbf{r} \leftarrow \mathbf{r} \setminus \{\boldsymbol{e}_{j+1}, \cdots, \boldsymbol{e}_{j+k-1}\}$
  end
  $j \leftarrow j + k - 1$
end

Algorithm 3: Angular point merge.

it is ensured that the VFOV is still valid and located fully inside the environment.

3.2.2 *Visibility Function*

The cleaned VFOV-polygons of the SPs build the foundation, for all SP-WPN visibility calculations. These binary relations state whether an SP can be connected with a WPN by a straight line that is entirely inside its FOV and does not intersect any obstacle, mountable or wall. The calculation can be simplified with a point in polygon test against the VFOV of the SP. A WPN that is within or on the boundary of the VFOV polygon of an SP is visible to that SP and vice versa as shown in Figure 13. Mathematically, it is stated using the notation of a binary visibility function (VF) as,

$$v(i,j) = \begin{cases} 1, & \text{if } \boldsymbol{w}_j \sqsubseteq \Psi_i, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

It can be efficiently calculated by using point in polygon test. The complexity of such test is $\mathcal{O}(m|E_\Psi|)$, whereas m is the number of WPNs and $|E_\Psi|$ is the number of VFOV edges (Shimrat, 1962). In conclusion, the worst case complexity of the VF calculation for n sensors is $\mathcal{O}(mn|E_\Psi^{\max}|)$, whereas $|E_\Psi^{\max}|$ is the max number of VFOV edges of all sensors.

Data: A VFOV polygon $P^i = (\mathbf{p}_1, \ldots, \mathbf{p}_n), n \geqslant 4$
where $\mathbf{p}_1$ is the SP that may contain a spike
An angular spike range $\mathfrak{s}$
Result: A polygon P^o removed of all spikes

$\mathfrak{P}^{env} \leftarrow \mathfrak{P}^{obs} \cup \mathfrak{P}^{wall} \cup \mathfrak{P}^{mnt}$
$E \leftarrow \{(\mathbf{p}_i, \mathbf{p}_{i+1}) \mid \mathbf{p}_i \in P^i\}$
$\mathbf{p}_i \leftarrow \mathbf{p}_2, \quad \mathbf{p}_{i+1} \leftarrow \mathbf{p}_3, \quad \mathbf{p}_{i+2} \leftarrow \mathbf{p}_4$
while $\mathbf{p}_{i+2} \neq \mathbf{p}_n$ **do**
  $d^{max} \leftarrow \max\{|(\mathbf{p}_1, \mathbf{p}_i)|, |(\mathbf{p}_1, \mathbf{p}_{i+1})|, |(\mathbf{p}_1, \mathbf{p}_{i+2})|\} \tan(\mathfrak{s})$
  $\mathbf{p}_i \leftarrow \mathbf{p}_{i+1}, \quad \mathbf{p}_{i+1} \leftarrow \mathbf{p}_{i+2}, \quad \mathbf{p}_{i+2} \leftarrow \mathbf{p}_{i+3}$
  `/* Check and handle mergeable spike. */`
  if $\|(\mathbf{p}_i, \mathbf{p}_{i+2})\| \leqslant d^{max}$ **then**
    $E \leftarrow E \setminus \{\boldsymbol{e}_i, \boldsymbol{e}_{i+1}\}$
    $E \leftarrow E \cup (\mathbf{p}_i, \mathbf{p}_{i+2})$
    $\mathbf{p}_i \leftarrow \mathbf{p}_{i+2}, \quad \mathbf{p}_{i+1} \leftarrow \mathbf{p}_{i+3}, \quad \mathbf{p}_{i+2} \leftarrow \mathbf{p}_{i+4}$
  end
  `/* Check and handle left spike. */`
  else if $\|\mathbf{p}_i, \boldsymbol{e}_{i+1}\| \leqslant d^{max}$ **then**
    Calculate nearest point $\mathbf{p}_h$ on line $\boldsymbol{e}_{i+1}$
    if $\emptyset = \{(\mathbf{p}_i, \mathbf{p}_h) \sqcap P \mid P \in \mathfrak{P}^{env}\}$ **then**
      $E \leftarrow E \setminus \{\boldsymbol{e}_i, \boldsymbol{e}_{i+1}\}$
      $E \leftarrow E \cup \{(\mathbf{p}_i, \mathbf{p}_h), (\mathbf{p}_h, \mathbf{p}_{i+2})\}$
      $\mathbf{p}_i \leftarrow \mathbf{p}_h, \quad \mathbf{p}_{i+1} \leftarrow \mathbf{p}_{i+2}, \quad \mathbf{p}_{i+2} \leftarrow \mathbf{p}_{i+3}$
    end
  end
  `/* Check and handle right spike. */`
  else if $\|\mathbf{p}_{i+2}, \boldsymbol{e}_i\| \leqslant d^{max}$ **then**
    Calculate nearest point $\mathbf{p}_h$ on line $\boldsymbol{e}_i$
    if $\emptyset = \{(\mathbf{p}_h, \mathbf{p}_{i+2}) \sqcap P \mid P \in \mathfrak{P}^{env}\}$ **then**
      $E \leftarrow E \setminus \{\boldsymbol{e}_i, \boldsymbol{e}_{i+1}\}$
      $E \leftarrow E \cup \{(\mathbf{p}_h, \mathbf{p}_{i+2}), (\mathbf{p}_i, \mathbf{p}_h)\}$
      $\mathbf{p}_i \leftarrow \mathbf{p}_h, \quad \mathbf{p}_{i+1} \leftarrow \mathbf{p}_{i+2}, \quad \mathbf{p}_{i+2} \leftarrow \mathbf{p}_{i+3}$
    end
  end
end
Check and handle left and mergeable spike for
$\mathbf{p}_i = \mathbf{p}_1, \quad \mathbf{p}_{i+1} = \mathbf{p}_2, \quad \mathbf{p}_{i+2} = \mathbf{p}_3$
Check and handle right and mergeable spike for
$\mathbf{p}_i = \mathbf{p}_{n-1}, \quad \mathbf{p}_{i+1} = \mathbf{p}_n, \quad \mathbf{p}_{i+2} = \mathbf{p}_1$.
$P^o \leftarrow \{v \mid v \in E\}$

Algorithm 4: Angular spike merge.

3.3 PAIRWISE SENSOR COMBINATIONS

The purpose of the placement models in this thesis is to provide a sensor positioning that satisfies task constraints in a given environment. Apart from the exact metric used to define the task constraints, a fundamental assumption is that positioning at a WPN can only be performed if it is covered by at least two sensors, also called a pairwise sensor combination (SC). To limit the complexity of the models within this thesis, the quality at a WPN will only be calculated for all SCs that "cover" it. These are all unique combinations of two SPs, which are visible from the WPN. Hence, the quality metric is restricted to be a function of exactly two SPs and a WPN.

To prepare the calculation of quality values, the SCs that cover a WPN have to be determined for every WPN by using the geometric information of the environment or the VF. Using the geometric environment information, all SCs that cover a WPN can be calculated by intersecting every combination of two VFOV polygons and then testing it against all WPNs. Nevertheless, an SC does not necessarily have an intersection polygon or the intersection polygon exist but no WPNs are sampled within the intersecting region as shown in Figure 13. Hence, an SC is only valid, if its VFOV intersections exists and cover one or more WPNs.

Beside the pure geometrical computation, the valid SCs can also be calculated using the VF as

$$c(\mathbf{s}_h, \mathbf{s}_i) = \begin{cases} 1, & \text{if } \exists \boldsymbol{w}_j \text{ such that } v(\mathbf{s}_h, \boldsymbol{w}_j) v(\mathbf{s}_i, \boldsymbol{w}_j) > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

All valid SCs of an environment will be denoted by a family of sets

$$\mathfrak{C} = \{\{\mathbf{s}_h, \mathbf{s}_i\} \mid (\mathbf{s}_h \in S) \wedge (\mathbf{s}_i \in S) \wedge (\mathbf{s}_h \neq \mathbf{s}_i) \wedge (c(\mathbf{s}_h, \mathbf{s}_i) = 1)\}. \tag{12}$$

In addition

$$\{h, i\} \coloneqq \{\mathbf{s}_h, \mathbf{s}_i\} \tag{13}$$

will be used as a short notation for the $C \in \mathfrak{C}$.

The number of operations to calculate all combinations for a given number of SPs and WPNs is the same for both approaches. Given m WPNs and n SPs, to test every of the

$$\frac{n^2 - n}{2}$$

unique combinations against all WPNs has a complexity of

$$m \frac{n^2 - n}{2} \Rightarrow \mathcal{O}(mn^2) \tag{14}$$

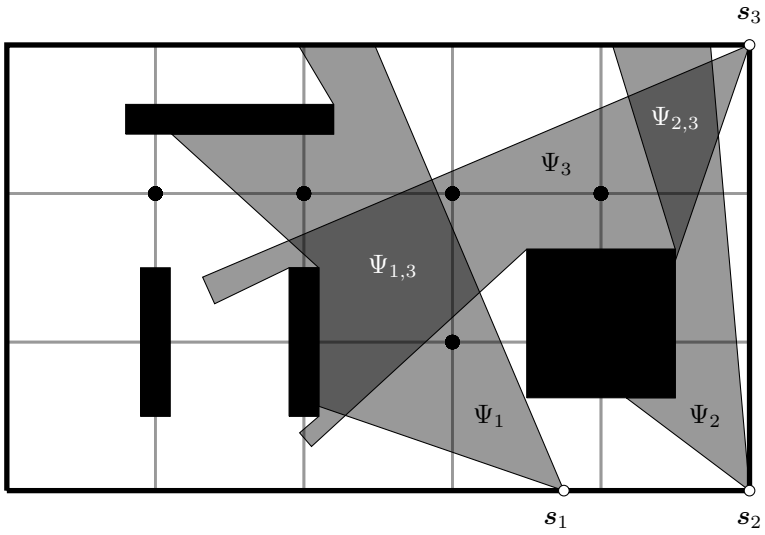


Figure 13: The intersection of three VFOVs of sensors $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$. Whereas the intersections $\Psi_{1,3}$ and $\Psi_{2,3}$, which are colored in dark gray are not containing any WPN, there is no VFOV intersection of $\mathbf{s}_1$ and $\mathbf{s}_2$. Hence, neither of the existing overlaps is valid with respect to Equation 11.

necessary operations. The difference is made by the kind of operation that has to be computed, where the geometric approach is based on calculation of a polygon intersection along with at least one point in polygon test for a valid SC, the VF based calculation only needs a single multiplication if all possible values for the VF have been precalculated.

3.4 FIELD OF VIEW OVERLAPPING

The validity of an SC does not provide any information about the quality at the WPNs it covers. An exception are SCs with SPs that are placed at the same sensor position (SPN). To provide a flexible selection, usually an angular resolution is chosen so the VFOV of SPs at the same SPN overlap. However, it is possible to select two sensors that have an overlapping FOV as shown in Figure 14.

The positioning quality at a WPN that is covered by two SPs placed at the same SPN will always be minimal, because the second sensor does not add any new information. Thus, if two sensors at the same SPN are to be selected their overlap should be minimized.

The problem of sensor overlapping is shown in Figure 14. For a mathematical description of the problem, let $\mathbf{s}_h$ and $\mathbf{s}_i$ be two SPs with the VFOVs Ψ_h and Ψ_i, an overlap of them exists if both sensors

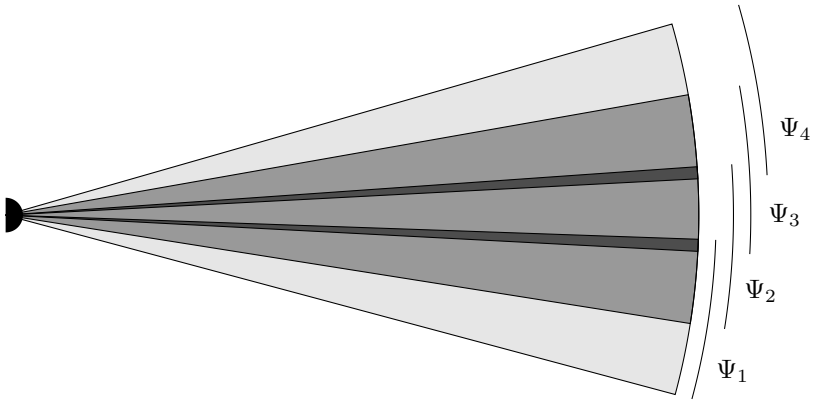


Figure 14: Four sensor VFOVs based on the same SPN. Due to the overlap, it would not make sense to choose the SP with the VFOV Ψ_1 and Ψ_2, since choosing the combination Ψ_1 and Ψ_3 would cover a greater area due to the smaller overlap. The same is true for the VFOV combination of Ψ_4 and Ψ_3 due to the smaller overlap of Ψ_2.

are at the same position ($x_h = x_i \wedge y_h = y_i$), and both VFOV intersect. The function that is used to calculate an overlap is defined as

$$f^o(\mathbf{s}_h, \mathbf{s}_i) = \begin{cases} 1, & \text{if } (x_h = x_i) \wedge (y_h = y_i) \wedge (\Psi_h \sqcap \Psi_i \neq \emptyset) \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

Using the overlap, it is possible to describe an avoidable overlap: two SPs that should not be selected together. Let $\mathbf{s}_h$, $\mathbf{s}_i$ and $\mathbf{s}_x$ be three SPs and Ψ_h, Ψ_i and Ψ_x their VFOVs, the function to calculate an avoidable overlap is defined as

$$a(\mathbf{s}_h, \mathbf{s}_i) = \begin{cases} 1, & \text{if } \exists \mathbf{s}_x \in S \text{ such that} \\ & \quad ((((\|\Psi_x \sqcap \Psi_i\| \leqslant \|\Psi_h \sqcap \Psi_i\|) \wedge \\ & \quad\quad (\Psi_h \sqsubseteq [\Psi_x \sqcup \Psi_i]))\vee \\ & \quad\quad (((\|\Psi_x \sqcap \Psi_h\| \leqslant \|\Psi_h \sqcap \Psi_i\|) \wedge \\ & \quad\quad\quad (\Psi_i \sqsubseteq [\Psi_x \sqcup \Psi_h])))\wedge \\ & \quad\quad (f^o(\mathbf{s}_x, \mathbf{s}_i) = 1) \wedge (f^o(\mathbf{s}_x, \mathbf{s}_h) = 1), \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

It is only one if there exist an SP $\mathbf{s}_x$ with a VFOV that:

- intersects the VFOV of both SPs $\mathbf{s}_h$ and $\mathbf{s}_i$ ($(f^o(\mathbf{s}_x, \mathbf{s}_i) = 1) \wedge (f^o(\mathbf{s}_x, \mathbf{s}_h) = 1)$),
- covers with one of the two polygons $\mathbf{s}_h$ or $\mathbf{s}_i$ the area of the other one ($\Psi_h \sqsubseteq [\Psi_x \sqcup \Psi_i]$ or $\Psi_i \sqsubseteq [\Psi_x \sqcup \Psi_h]$),
- forms an intersecting polygon with one of $\mathbf{s}_i$ or $\mathbf{s}_h$ that has a size less than the size of the intersection of $\mathbf{s}_i$ and $\mathbf{s}_h$ ($\Psi_i \sqsubseteq [\Psi_x \sqcup \Psi_h]$ or $\|\Psi_x \sqcap \Psi_i\| \leqslant \|\Psi_h \sqcap \Psi_i\|$).

In essence, this means that two sensors should not be placed at the same position if they overlap and there is another sensor that has a smaller overlap. In Figure 14, there is an avoidable positive overlap of SPs 2 and 1, represented by their VFOV Ψ_2 and Ψ_1. Since the VFOV Ψ_3 of sensor 3 also overlaps Ψ_1 but has a greater φ then Ψ_2.

The short form of the function is

$$a(h,i) \coloneqq a(\mathbf{s}_h, \mathbf{s}_i).$$

3.5 POSITIONING QUALITY METRICS

The definition of avoidable overlaps is a simple construct to prohibit sensor combinations that do not add any positioning quality. Beside this quality criterion, which only depends on the SPNs, in general the positioning quality has to be measured at a WPN. Thus, a quality metric for positioning applications is function that expresses how the quality is influenced by the geometric relation of one or more visible sensors to a position in the workspace.

3.5.1 *Single Sensor Quality*

Given a positioning system based on triangulation and sensors that induce an angular error for bearing measurements, the error of a position estimate scales with the distance of the sensors from the target. Therefore, the quality of the positioning increases when the distance between the target and the sensors decreases. This simple relationship can be used to serve as a quality measure that only depends on a WPN and a single SP. The advantage of such a quality metric is its simplicity, as it does not require any SC to be computed. Nevertheless, it cannot provide any information about the positioning quality possible at a WPN for triangulation or trilateration based positioning systems.

Such a quality function that calculates a quality measure in the range of $[0,1]$ may be defined as

$$f^{qsw}\left(\mathbf{s}_i, \boldsymbol{w}_j\right) = \left(1 - \left(\frac{\|\mathbf{s}_i, \boldsymbol{w}_j\|}{r}\right)\right) v(\mathbf{s}_i, \boldsymbol{w}_j). \tag{17}$$

It calculates the distance between the SP and the WPN ($\|\mathbf{s}_i, \boldsymbol{w}_j\|$). Dividing it by the maximum sensing range r ensures that the result at the maximum sensing range is one. Using

$$1 - \left(\frac{\|\mathbf{s}_i, \boldsymbol{w}_j\|}{r}\right)$$

sets the maximum quality (1) to be at the SP and the minimum quality (0) to be at the maximum sensing range. The term $v(i,j)$ ensures that the quality outside the sensing range is also zero.

A single sensor quality can also be derived from the sensor properties. Some sensor systems—like thermopile arrays—are more sensitive to signals originating in the middle of their FOV than to ones originating at the sides, which might be expressed in a similar manner. Further on, single sensor quality metrics are included in (Kirchhof, 2013).

3.5.2 *Sensor Pairwise Quality*

For simple 2d triangulation and trilateration measurement models, where two sensors $\mathbf{s}_h$ and $\mathbf{s}_i$ measure the orientation/distance to a target $\boldsymbol{w}_j$ Kelly (Kelly, 2003) derived the Geometric Dilution of Precision (GDOP) as "a Jacobian determinant that expresses the scalar multiplier which converts a differential volume in pose space to its corresponding differential volume in measurement space". Thus, in essence the GDOP is a metric that expresses the uncertainty of position estimation based on a given SC-WPN configuration and defines how much the measurement error is magnified by the geometric relation of the sensor system (Torrieri, 1984). A positioning quality metric in this thesis is defined to be the inverse of the GDOP function that is restricted to the range $[0, 1]$.

Let γ_{hij} be the inner bearing angle of this configuration, the GDOP for angular measurements is defined as

$$u(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}) = \frac{\|\mathbf{s}_h, \boldsymbol{w}\| \|\mathbf{s}_i, \boldsymbol{w}\|}{\sin(\gamma_{hij})}. \tag{18}$$

For distance measurements the GDOP is

$$u^d(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}) = \frac{1}{\sin(\gamma_{hij})}. \tag{19}$$

In essence, it is a function that values how "good" the intersection of the uncertainty polygons of two SPs will be at the WPN. Kelly (Kelly, 2003) derived that the measurement error is independent of the measured distance, if the positioning system is based on trilateration. It only depends on the inner bearing angle of the measured point in relation to the two sensors. In Figure 15, the inner bearing angle is plotted in relation to the polygon area and boundary of the uncertainty polygon. It can be seen that the GDOP is also a good representation of the polygon sizes except for inner bearing angles $\geqslant 140°$. Here, the GDOP rapidly increases, whereas the polygon area and boundary length slightly decreases. This behavior is due to the uncertainty circles that barely overlap at these angles and therefore get smaller again. Thus, if the object can be detected by both of the sensors the overlap of the uncertainty regions is quite small as shown in Figure 16. In contrast, the uncertainty circles at an inner bearing angle of $0°$ are maximal because both sensors have to be at the same position and therefore the uncertainty is the highest.

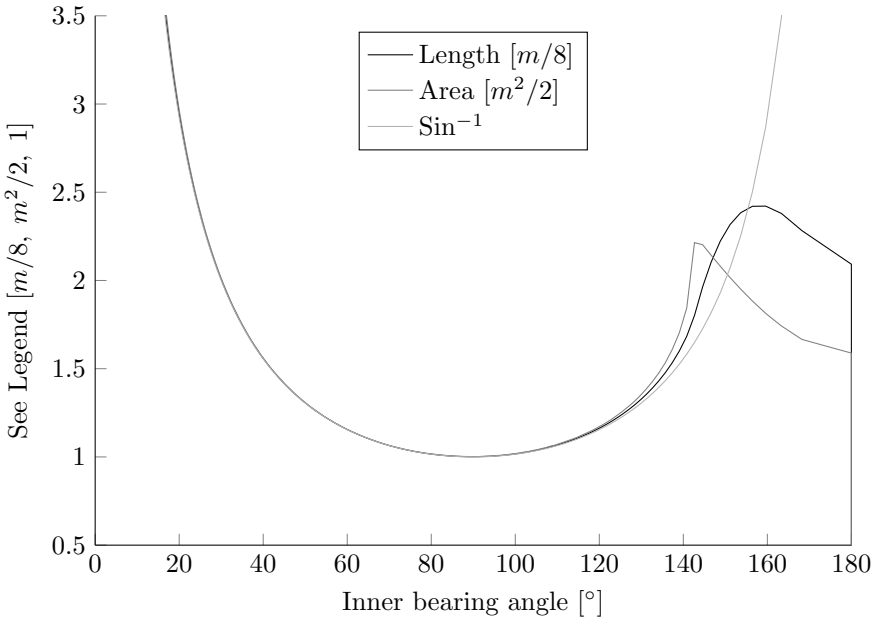


Figure 15: The polygon area, boundary length and the inverse of the sine for inner bearing angles in range of $[0, 180]$. The area and boundary length are transformed by a linear scale factor to match the inverse sine.

The uncertainty circles for triangulation based configurations are comparable. An inner bearing angle of 0° means that the uncertainty triangles have the same orientation, and when they are at the same position, the area of the uncertainty intersection polygon is maximal. For a bearing angle of 180° in turn, the real object position can be somewhere between both sensors.

The optimal inner bearing angle is 90°. However, contrary to the trilateration scenario, the size of the intersection polygon in a triangulation scenario is also related to the distance from both sensors, as stated in Equation 18. The equation uses absolute lengths, which makes the uncertainty values dependent on the used measurement unit. To state a generalized GDOP, the limited detection range of the sensors (r) is exploited to normalize the distance. Let $\mathbf{s}_i$ be an SP and $\boldsymbol{w}_j$ be a WPN, the distance

$$d_{i,j} = \|\mathbf{s}_i, \boldsymbol{w}_j\|$$

is normalized by dividing the absolute distance by the maximum detection range of the sensor as

$$d^n_{i,j} = \frac{d_{i,j}}{r}.$$

Applying this distance transformation to the GDOP equation for triangulation (18) leads to

$$u\left(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}_j\right) = \frac{d^n_{i,j} \cdot d^n_{h,j}}{\sin(\gamma_{hij})}. \tag{20}$$

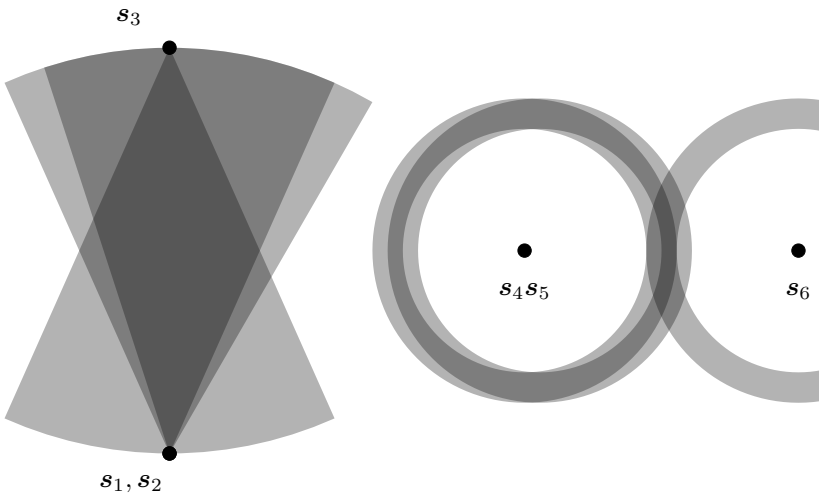


Figure 16: An exemplary visualization of the triangulation and trilateration uncertainty regions for angles close to 0° and 180°.

The values of the functions are a measure of the uncertainty at a defined point. They are $\geqslant 0$, whereas 0 represents the least possible uncertainty. A problem with this function is that it does not represent the limited detection range of a sensor. If a WPN is not in range of one of the sensors, the GDOP does not exist.

Two approaches can be used to handle this case. The first one is to use the GDOP only for points that are verified to be in the range of both sensors. The second one is to include the detection range in the GDOP function. The straightforward approach is to use a piecewise function that returns a maximum uncertainty value u^{max} if the distance to one of the SPs is outside the detection range as

$$u^r(\mathbf{s}_i, \mathbf{s}_h, \boldsymbol{w}_j) = \begin{cases} u^{max}, & \text{if } d^n_{i,j} \text{ or } d^n_{h,j} > 1, \\ u\left(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}\right), & \text{otherwise.} \end{cases}$$

In essence, the maximum uncertainty values state that a certain quality is achievable at a WPN. This relation can also be directly expressed by transforming the maximum uncertainty, for which lower values indicate a higher positioning quality, into a minimum quality function, for which a higher outcome indicates a higher quality.

To express the positioning quality at a WPN, the GDOP can be transformed by applying an upper bound of one and an appropriate scale q^{scale} to the quality values as

$$q^u = 1 - q^{scale} u\left(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}_j\right). \tag{21}$$

The purpose of this transformation is to limit the quality values to the range of $[0, 1]$, whereas 0 represents the worst and 1 represents the best possible quality as,

$$q(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}_j) = \min\{1 - q^{scale} \frac{d^n_{i,j} \cdot d^n_{h,j}}{\sin(\phi)}, 0\}. \tag{22}$$

Beside the full notation, the short form

$$q(h,i,j) \coloneqq q(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}_j) \tag{23}$$

will be used throughout the thesis. Furthermore, instead of the two SPs, the function is used with a reference to an SC as

$$q(c,j) \coloneqq q(h,j,i), \quad j \in C_c, h \in C_c, j \neq h. \tag{24}$$

In addition, the function will be used to calculate the qualities whenever it can be ensured that $d^n_{i,j}, d^n_{h,j} \leqslant 1$. Otherwise, the quality is calculated using the piecewise function that includes the detection range

$$q^r\left(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}_j\right) = \begin{cases} 0, & \text{if } (d^n_{i,j} > 1) \vee (d^n_{i',j} > 1), \\ q(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}_j), & \text{otherwise.} \end{cases} \tag{25}$$

In summary, the quality metric can be seen as a function approximation of uncertainty polygon properties. The overall excellent representation of the polygon size makes the GDOP a well-suited metric on which an SPP can be stated and calculated. Its advantage is that in relation to the polygon parameters it is computational inexpensive to calculate. The choice of exploiting the restricted sensing range to transform the GDOP into a function with a defined output range provides the opportunity to state SPPs independently of the used measurement unit. In addition, the defined output range gives the opportunity to embed the visibility calculations within the function values. The representation of the GDOP as a function that returns minimum qualities instead of maximum uncertainties mainly serves illustration purposes and both can be used interchangeably.

Independent from the representation of the GDOP, working with this metric implies a task specific definition of acceptable values that satisfy the applicants' requirements on the positioning system that is to be deployed. A sample definition of the quality function using the parameters of the ThILo system is given in Section 7.1.

4 DISCRETE MODELS

The first approach on finding optimal solutions to the sensor placement problem is to use a discretized environment description and optimize for the number of placed SPs. Therefore, in this chapter it is shown how a simple binary integer coverage model can be stated based on the preprocessed variables (Section 4.1). This model is extended with constraints that represent the pairwise sensor quality metric (Section 4.2) and finally transformed into a multi objective model that will always result in the solution with the fewest sensors and the maximum overall quality (Section 4.3). The models in this chapter have already been published by Kirchhof (Kirchhof, 2013).

4.1 SIMPLE k-COVERAGE MODEL

As a first step towards models that enable minimum positioning quality, simple k-Coverage problems are presented. Here, k stands for the number of different observers/sensors that are to cover a position in the environment. For the sensor placement problem (SPP), this is the number of SPs that cover a WPN. Thus, for $k = 1$ every WPN will be covered by one SP. For $k \geqslant 2$, solving the k-coverage problem provides a basis for positioning applications because every WPN has to be covered by at least two SPs. For $k = 2$, a BIP model that models the k-coverage problem will be called Simple Two-Coverage Model (STCM).

It can be stated using a binary decision vector $\mathbf{x}^s$. Its elements define whether an SP is to be selected. Thus,

$$x_i^s = \begin{cases} 1 & \text{if } \mathbf{s}_i \text{ is placed,} \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

In addition, by using avoidable overlaps (Equation 16) and the VF $v(i,j)$ (Equation 10), the STCM is the following:

$$\text{minimize} \quad \sum_{\forall i} x_i^s \tag{27}$$

$$\text{subject to} \quad \sum_{\forall i} x_i^s \, v(i,j) \;\; \geqslant 2, \;\; \forall\, j \tag{28}$$

$$x_i^s \sum_{\forall h} a(h,i) \;\; = 0, \;\; \forall\, i \tag{29}$$

Here, objective 27 minimizes the number of sensors to be placed. Constraint 28 states that every WPN has to be covered by at least two

sensors that are selected in $\mathbf{x}^s$ and constraint 29 represents the avoidable overlap. It prohibits that two sensors, placed at the same position have a large overlap, which in turn might cover WPNs with an insufficient quality (see Section 3.5.2). Therefore, the avoidable overlap constraint increases the quality of the found solution.

Optionally, a cost vector $\mathbf{c}^s$ that weights the importance of the SPs can be subtracted from the objective function. If some SPs are to be preferred, their weight can be included in the objective, which in turn changes to:

$$\text{minimize} \quad \sum_{\forall i} c_i^s x_i^s. \tag{30}$$

Prioritizing sensor positions is always an additional option that can be used to bias optimizations towards a cost effective solution. It is useful if some SPs can be equipped cheaper and thus, are preferable over others. Nevertheless, within the scope of this thesis, such constraints are not taken into account.

4.2 MINIMUM SENSOR PAIRWISE QUALITY MODEL

Due to the fact, that the Simple Two-Coverage Model (STCM) does not include any quality criteria other than the sameplace constraints, the solution is likely to be far from optimal for the SPP stated in Section 2.2. To solve the SPP, the SP-SP-WPNs qualities have to be modeled, which is done using a BIP model, namely the MSPQM. The MSPQM can be used to optimally solve the SPP in a discrete domain.

The section presents two separate ways are shown to model the qualities. The strict way that ensures one SC provides a minimum quality at each covered WPN and the relaxed way where the quality criterion at a WPN may be fulfilled by a summation of the qualities from all SCs that cover it. For both approaches, a mapping of two SPs to an SC has to be included in the optimization model. Therefore, a binary decision vector $\mathbf{x}^z$ is defined for the SCs with elements

$$x_c^z = \begin{cases} 1, & \text{if SC c is selected} \\ 0, & \text{otherwise} \end{cases} . \tag{31}$$

It can be mapped to the sensor decision vector $\mathbf{x}^s$(Equation 26) using the three constraints

$$\begin{aligned} x_i^s - x_c^z &\geqslant 0 \\ x_h^s - x_c^z &\geqslant 0, \quad \forall (C_c = \{h, i\}) \in \mathfrak{C} \\ x_i^s + x_h^s - x_c^z &\leqslant 1 \end{aligned} \tag{32}$$

This is an equivalence mapping, since the inequalities force x_i^s and x_h^s to be one if x_c^z is selected and the other way round (Kallrath, 2013).

In addition, the selection $\forall(C_c = \{h, i\}) \in \mathfrak{C}$ defines that the mapping is applied only to valid SCs and that the c^{th} element of the family of SCs sets $\mathfrak{C}$ is mapped to the two SPs it holds, the h^{th} and i^{th} SP of the set of SPs S. As a short form of this

$$\forall C_c$$

is used for the rest of this chapter.

The strict approach to model quality constraints is to perform a quality decision upfront and test if the quality at a WPN with respect to an SC is above a predefined threshold. For this, the quality function $q(c, j)$ (Equation 24) is discretized to a binary quality function

$$q^b(c, j) = \begin{cases} 1, & \text{if } q(c, j) \geqslant q, \\ 0, & \text{otherwise.} \end{cases} \tag{33}$$

Using the binary of all SC-WPN qualities, both decision vectors ($\mathbf{x}^s$, $\mathbf{x}^z$) and their mapping, the quality constraints can be included in the optimization model as

$$\begin{aligned}
\text{minimize} \quad & \sum_{\forall i} x_i^s && && (34)\\
\text{subject to} \quad & \sum_{\forall c} x_c^z q^b(c, j) && \geqslant 1, \ \forall j && (35)\\
& x_h^s - x_c^z && \geqslant 0, \quad \forall C_c && (36)\\
& x_i^s - x_c^z && \geqslant 0, \quad \forall C_c && (37)\\
& x_i^s + x_h^s - x_c^z && \leqslant 1, \quad \forall C_c && (38)\\
& x_i^s \sum_{\forall h} a(h, i) && \leqslant 0, \quad \forall i && (39)
\end{aligned}$$

Here, constraint 35 enforces the selection of at least one SC with a sufficient quality for each workspace point. The constraints 36 and 37 map the sensor combination decision vector back to the single sensor decision vector and 38 ensures that the additional combinations which arise among the SPs when selecting multiple SPs are represented by a selection of the respective SCs. Finally, constraint 39 ensures that the avoidable overlap is minimized. Altogether, the strict MSPQM ensures that one SC for each WPN provides the minimum quality.

In contrast to the strict MSPQM, the relaxed MSPQM uses the SC-WPN qualities directly. Therefore, constraint 35 is changed to

$$\sum_{\forall c} x_c^z q(c, j) \ \geqslant q, \quad \forall j. \tag{40}$$

This constraint enforces the selection of SCs whose quality is in sum greater than the minimum quality. Therefore, a WPN can also be covered using only SCs that do not fulfill the minimum quality constraint.

The underlying assumption is that the information gain from more sensors also accounts for better positioning quality.

In general, it cannot be determined if the positioning quality would be better at a certain WPN or the environment if there are few low quality information instead or one with high quality. To decide which of the two options provides a better positioning quality, the geometric properties of the SPs and the sensor properties would have to be taken into account and evaluated at every WPN. Since this is not part of the scope of this thesis, the decision is referred to the architect of a positioning system. In the scope of the evaluations, only the strict MSPQM is evaluated. For further results, the interested reader is referred to Kirchhof (Kirchhof, 2013).

4.3 BEST SENSOR PAIRWISE QUALITY MODEL

The MSPQM will find the solution with the least sensors that cover the workspace with a desired quality at every WPN. If there is more than one valid solution with the minimum possible number of sensors, each one of them is as good as the others from the model's view. Thus, if the optimization goal is not only to find the solution with the lowest number of sensors but also to ensure to get the solution with the highest quality, the latter has to be included in the model.

A problem with this is the definition of the "best" solution. Since all visible WPNs of every SC provide quality values, the best solution may be the one that maximizes the overall sum of every available quality, the mean of maximum quality values at each WPN, the minimum of maximum values at each WPN and so forth. In respect to the ability to include a solution weighting into the model, the BSPQM will model the problem to use the fewest sensors with the highest sum over all available qualities.

To achieve this, all SC-WPN qualities have to be included into the model, which can be performed by adapting the objective function in two steps. The first one is to define an additional continuous variable q_j^{wpn} for each WPN w_j. These variables are added to the relaxed model by adding the constraint

$$-q_j^{wpn} + \sum_{\forall c} x_c^z \, q(c,j) \; = 0, \quad \forall \, j. \tag{41}$$

Here, each q_j^{wpn} represents the overall quality at a WPN j that can be achieved with the currently selected sensors. To find the solution with the best overall quality, the newly created variables have to be

included in the objective function. In addition, a constant σ is multiplied to the qualities to scale their influence.

$$\text{minimize} \quad \sum_{\forall i} x_i^s - \sigma \sum_{\forall j} q_j^{wpn} \tag{42}$$

$$\text{subject to} \quad -q_j^{wpn} + \sum_{\forall c} x_c^z q(c,j) = 0 \quad \forall j. \tag{43}$$

$$\sum_{\forall c} x_c^z q^b(c,j) \geqslant 1, \quad \forall j \tag{44}$$

$$x_h^s - x_c^z \geqslant 0, \quad \forall C_c \tag{45}$$

$$x_i^s - x_c^z \geqslant 0, \quad \forall C_c \tag{46}$$

$$x_i^s + x_h^s - x_c^z \leqslant 1, \quad \forall C_c \tag{47}$$

$$x_i^s \sum_{\forall h} a(h,i) \leqslant 0, \quad \forall i \tag{48}$$

In essence, the model structure is comparable to the MSPQM because the constraints 44 to 48 are still necessary to state the decision vector mapping and the restricted minimum quality constraint. The additional constraint 43 maps the sum of quality values to the continuous variables q_j^{wpn}. These variables are also part of the objective function and add a secondary goal: to find the solution with the best possible quality. Therefore, the model states a multi criteria optimization problem and the scale variable σ is necessary to restrain the influence of the quality maximization objective.

To determine a value for σ, it has to be considered that every selected SC c increases $\sum_j q_j^{wpn}$ by

$$\sum_{\forall j} q(c,j).$$

Since the quality is defined in the range $[0,1]$ and let $|W|$ be the number of all WPNs then

$$\sum_{\forall j} q(c,j) \leqslant |W|.$$

Therefore, an SC with $\sum_{\forall j} q(c,j) > 2$ would always be selected because the SC decreases the objective function, since the additional cost for its selection is $2 - q^{wpn}$, which is < 0 for $q^{wpn} > 2$. For sensors that form an SC with already selected sensors, the situation is even worse because their additional cost is only one.

In practice, the objective function decrease introduced by

$$\sum_{\forall j} q(c,j)$$

is usually not as high as the total number of WPNs $|W|$ because no SC has the maximum quality at every WPN. Nevertheless, because

$|W| \gg 10$ in real world scenarios, the decrease $\mathbf{q}^{\text{wpn}}$ is usually $\gg 2$ for selected SCs. To ensure that the quality decrease of the objective function for an additionally selected SC is smaller than the increase that is introduced by the newly selected SPs, the trivial way is to choose σ as the sum over all qualities as

$$\sigma = \left(\sum_{\forall c, \forall j} q(c, j) \right)^{-1} . \tag{49}$$

This is an overestimation, since it ensures that the decrease in quality value is always $\ll 1$ if an SC is added. Only if all sensors would be selected the decrease would be one. Due to the small value increase, this approach is not suited for real world solvers. A better way of approximating σ is to compute the maximum quality decrease that is possible for a single sensor by using

$$\mathfrak{C}(h) = \{C_c \mid (C_c \in \mathfrak{C}) \wedge (C_c \cap \{h\} \neq \emptyset)\} \tag{50}$$

as a selection on $\mathfrak{C}$ that only includes SCs in which h is present. With this selection, the inverse of the maximum overall quality increase that can be achieved when selecting a sensor can be calculated as

$$\sigma = \left(\max_{h} \sum_{\forall C_c \in \mathfrak{C}(h)} \sum_{\forall j} q(c, j) \right)^{-1} , \tag{51}$$

which is the sum of all quality values that sensor takes part in.

In conclusion, the BSPQM provides the possibility to calculate the solution with the lowest number of SPs as its first optimization criterion and the best overall sum of qualities as its second optimization criterion. By adequately scaling the second criterion, it is ensured that it does not interfere with the first one.

5 APPROXIMATIONS

The proposed optimization models can be solved to find the global optimal solution for a discretized version of the SPPs. Their disadvantage is the rapid increase in runtime, which is shown by the large number of SPPs that could not be solved in reasonable time (see Section 7.4). This limits the maximum number of SPs and WPNs, if solutions are to be found in reasonable time.

To provide methods that allow the computation of valid solutions of the SPP for greater input sizes and therefore a higher level of discretization correctness, this chapter introduces adaptations of the common strategies approximation and problem partitioning. The goal of the first Sections 5.1 and 5.2 of this chapter is to present optimization models and heuristics with simplified goal functions (compared to the MSPQM) that allow to find solutions to the discretized SPP in reasonable time. In contrast to the MSPQM, which fully defines the SPP and whose global optimal solution is the best possible sensor placement in a discrete domain, the proposed methods are not supposed to optimally solve the SPP. Nevertheless, they produce valid solutions to the MSPQM and therefore, they will be referred to as "approximations" because they approximate the global optimal MSPQM (discrete SPP) solution.

At first, an approximate optimization model is presented and its WCAR in relation to the MSPQM is derived (Section 5.1). In addition, a search strategy is presented that can be used to improve approximate solutions. To provide possibilities to approximate the MSPQM in polynomial time complexity, two greedy heuristics are derived (Section 5.2). Finally, a strategy is introduced to separate the SPPs into smaller sub problems, which can be solved independently (Section 5.3).

5.1 SENSOR COMBINATION OPTIMIZATION

The first approximation technique is a simplification of the MSPQM. The idea is that instead of minimizing the number of sensors, the number of SCs is minimized. The approximation approach will be called Sensor Combination Optimization (SCO).

The problem is stated by using the binary decision vector x_c^z for every SC in $\mathfrak{C}$ and the binary SC-WPN quality function $q^b(c, j)$ (Equation 33) to

$$\text{minimize} \quad \sum_{\forall c} x_c^z, \tag{52}$$

$$\text{subject to} \quad \sum_{\forall c} x_c^z q^b(c, j) \geqslant 1, \ \forall j \tag{53}$$

The objective (Equation 52) is to minimize the number of SCs. The only constraint (Equation 53) states that every WPN is covered by at least one SC with a sufficient quality. In contrast to the MSPQM, this simple set covering problem is still NP-complete but can be solved efficiently (Yelbay, Birbil, & Bülbül, 2014).

To show the quality of this model in relation to a global optimal SPP solution of the MSPQM, the following section derives a worst case approximation ratio (WCAR). This is the worst-case ratio of an optimal SCO solution to an optimal MSPQM solution.

5.1.1 *Worst Case Approximation Ratio*

The worst case approximation ratio (WCAR) derivation follows an inductive scheme and is split over several lemmas. Its purpose is to show that general relations among solutions of the full discrete SPP MSPQM and the discrete SPP SCO, which is used as an approximation of the MSPQM, can be used to calculate a worst case approximation rate.

At first, some properties of an optimal MSPQM solution are stated.

> An optimal MSPQM solution $\mathfrak{Z}$ is the family of sets of SCs $C \in \mathfrak{C}$, that cover all WPNs. The solution minimizes the number of different SPs that are contained in the SCs, as shown in Section 4.2.

The solution can be achieved by solving the MSPQM with a BIP solver, which gives a vector x^s of binary variables that are one if the respective SP is selected. The vector can be transformed to a family of SCs sets by

$$\mathfrak{Z} = \{\{\mathbf{s}_h, \mathbf{s}_i\} \mid (x_h^s = 1) \wedge (x_i^s = 1) \wedge (x_h^s \neq x_i^s)\}.$$

The family of sets $\mathfrak{Z}$ contains all unique combinations $\{\mathbf{s}_h, \mathbf{s}_i\}$ for the selected sensors. In addition, let $\mathfrak{W}^Z$ be the family of sets that contains the covered WPNs for every SC in $\mathfrak{Z}$ as

$$\mathfrak{W}^Z = \left\{ \left\{ \boldsymbol{w}_j \mid q^b(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}_j) = 1 \right\} \mid \{\mathbf{s}_h, \mathbf{s}_i\} \in \mathfrak{Z} \right\}.$$

Thus, the flattened $\mathfrak{W}^Z$ must include every $\boldsymbol{w} \in W$ or

$$\bigcup \mathfrak{W}^Z = W$$

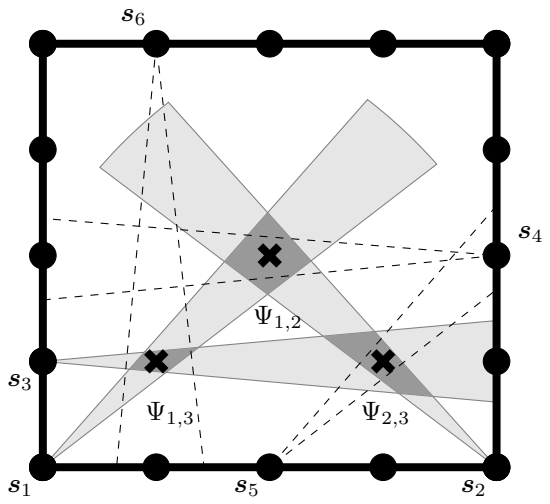


Figure 17: Three sensors s_1, s_2 and s_3 that cover the three WPNs with sufficient quality and provide an optimal MSPQM solution. For each sensor, the VFOVs Ψ_1 and Ψ_2 are plotted in light gray and the VFOV intersections $\Psi_{h,i} = \Psi_h \cap \Psi_h$ are dark gray. The dotted lines are the VFOVs of the sensors s_4, s_5 and s_6 that could also be used to cover the workspace points.

must hold for Z to be a valid solution. An example of an optimal MSPQM solution is the selection of s_1, s_2 and s_3 for the SPP shown in Figure 17.

In contrast to the optimal MSPQM solution, the properties for an optimal SCO solution are

> An optimal SCO solution $\mathfrak{A}$ is the family of sets of SCs $C \in \mathfrak{C}$, that cover all WPNs. It minimizes the number of different SCs that are contained in $\mathfrak{A}$.

The solution can be achieved by solving the SCO with a BIP solver, which gives a vector x^z of binary variables that are one if the respective SC is selected. The vector can directly be transformed to a SCO solution as

$$\mathfrak{Z}^a = \{C_c \mid (x_c^z = 1)\}.$$

In addition, let $\mathfrak{W}^A$ be the family of sets that contains the covered WPNs for every SC in $\mathfrak{A}$ as

$$\mathfrak{W}^A = \left\{ \left\{ w_j \mid q^b(C, j) = 1 \right\} \mid C \in \mathfrak{A} \right\}.$$

As for the MSPQM, the flattened $\mathfrak{W}^A$ must include every $w \in W$.

After the introduction of the optimal MSPQM and SCO solutions, their properties are explored.

Lemma 1. *Let W_u, W_v and W_w be three sets of WPNs from $\mathfrak{W}^A$ and let C_u, C_v, C_w be the three corresponding SCs. If $W_v \cup W_w \subset W_u$ the SCO will always select C_u.*

Proof. By contradiction. If the SCO would choose C_v, C_w the solution would have 2 instead of 1 SC and therefore would not be optimal. □

The Lemma also implies that every set of WPNs that is part of the optimal SCO solution has at least one unique WPN, otherwise all WPNs would already been covered and this set would not be part of the optimal solution.

Lemma 2. *Let W_u, W_v and W_w be sets of WPNs from $\mathfrak{W}^Z$ and let C_u, C_v, C_w be the three corresponding SCs. If $W_v \cup W_w \subset W_u$ the optimal MSPQM solution $\mathfrak{Z}$ will contain $W_v \wedge W_w$ if the number of SPs of $\mathfrak{Z} \cup \{C_v, C_w\}$ is less than the number of SPs of $\mathfrak{Z} \cup \{C_u\}$,*

$$|\bigcup(\mathfrak{Z} \cup \{C_v, C_w\})| < |\bigcup(\mathfrak{Z} \cup \{C_u\})|. \tag{54}$$

Proof. By contradiction. If $\{C_v, C_w\}$ are part of the optimal MSPQM solution and

$$|\bigcup \mathfrak{Z} \cup \{C_v, C_w\}| > |\bigcup \mathfrak{Z} \cup \{C_u\}|. \tag{55}$$

the solution would not be optimal. □

In essence, the Lemma states that if sensors are a definite part of the optimal MSPQM solution, additional combination of them do not increase the solutions fitness value. Thus, sensors are only part of the optimal MSPQM solution if they themselves are part of one SC that covers a WPN, which cannot be covered by any other combination of the sensors in $\mathfrak{Z}$.

Lemma 1 and 2 can be used to state the relationship of both optimal solutions.

Lemma 3. *Let $\mathfrak{W}^A$, $\mathfrak{A}$ define the optimal SCO solution and $\mathfrak{W}^Z$, $\mathfrak{Z}$ the optimal MSPQM solution to the same SPP. Their fundamental relations are*

$$|\bigcup \mathfrak{W}^A| = |\bigcup \mathfrak{W}^Z| \tag{56}$$

$$|\mathfrak{A}| \leqslant |\mathfrak{Z}| \tag{57}$$

$$|\bigcup \mathfrak{A}| \geqslant |\bigcup \mathfrak{Z}| \tag{58}$$

Proof. Since both are only optimal if they cover every WPN, Equation 56 is true by definition.

Equation 57 is true due to Lemmas 1 and 2. Since the SCO will not allow two SCs in the optimal solution if all of the WPNs they cover can be covered by any single SC. Thus, the SCO is always optimal with respect to the SCs, whereas the MSPQM is not.

Equation 58 is true since the MSPQM solution is sensor optimal and the SCO solution is not. □

From Lemma 3 an SPP can be derived that always leads to an optimal MSPQM and SCO solution.

Lemma 4. *Let $\mathfrak{C}$ be the family of sets of all valid SCs from an SPP. If and only if every SP is only part of one SC and therefore,*

$$C_c \cap C_d = \emptyset, \quad \forall C_c \in \mathfrak{C}, \forall C_d \in \mathfrak{C}, C_c \neq C_d \tag{59}$$

an existing solution to the SCO will be optimal for the MSPQM as well and vice versa.

Proof. The problem is stated in a way that a selection of SCs has to be made to cover the WPNs. In contrast, adding a single SP $\mathbf{s}_h$ will not cover any new WPNs. To cover WPNs, the SP $\mathbf{s}_i$, which is also part of the only SC $\mathbf{s}_h$ is contained in, must be added as well. □

To specify the general relationships among the optimal solutions, the relation between the number of SPs and SC has to be explored.

Lemma 5. *The highest SPs to SCs ratio in an optimal MSPQM or SCO solution is 2, thus*

$$|\bigcup \mathfrak{Z}| \leqslant 2 \cdot |\mathfrak{Z}|. \tag{60}$$

This is also true if $\mathfrak{Z}$ is replaced by $\mathfrak{A}$.

Proof. By reasoning. Since two SPs form an SC, and all SPs in an optimal solution have to belong to one or more SCs the ratio is two whenever one SP belongs to exactly one SC. □

Lemma 6. *The lowest possible SPs to SCs ratio in an optimal MSPQM or SCO solution is related to the number of SPs as*

$$|\mathfrak{Z}| \leqslant \frac{1}{2}\left(|\bigcup \mathfrak{Z}|^2 - |\bigcup \mathfrak{Z}|\right). \tag{61}$$

This is also true if $\mathfrak{Z}$ is replaced by $\mathfrak{A}$.

Proof. The lowest SPs to SCs ratio is given if the SCs contain any possible combination of two SPs. Therefore, the maximum number of SCs for a given number of SPs is the number of unique combinations of two SPs. □

The Lemma can be used to calculate the minimum number of SPs necessary to form the SCs of an optimal solution by transforming Equation 61 via

$$\begin{aligned} 2|\mathfrak{Z}| + \frac{1}{4} &\leqslant |\bigcup \mathfrak{Z}|^2 - |\bigcup \mathfrak{Z}| + \frac{1}{4} \\ \sqrt{2|\mathfrak{Z}| + \frac{1}{4}} &\leqslant |\bigcup \mathfrak{Z}| - \frac{1}{2} \end{aligned}$$

to

$$|\bigcup \mathfrak{Z}| \geqslant \frac{1}{2} + \sqrt{2|\mathfrak{Z}| + \frac{1}{4}}, \tag{62}$$

which is also true for $\mathfrak{A}$.

To combine the presented Lemmas and state a general worst case relation between the number of selected SPs of an optimal MSPQM solution to an optimal SCO solution for the same SPP two theorems are given:

Theorem 1. *If an optimal MSPQM solution is given, the WCAR $\mathfrak{h}^{oa}$ that states how good an optimal SCO can approximate the same SPP is given as*

$$\mathfrak{h}^{oa} = \frac{|\bigcup \mathfrak{A}|}{|\bigcup \mathfrak{Z}|}$$
$$\mathfrak{h}^{oa} \leqslant \frac{2|\mathfrak{A}|}{|\bigcup \mathfrak{Z}|} \leqslant \frac{2|\mathfrak{Z}|}{|\bigcup \mathfrak{Z}|}$$

Proof. Lemma 5 serves as an estimation of a solution with the same number of SCs but greater number of SPs, whereas Lemma 3 states that

$$|\mathfrak{Z}| \geqslant |\mathfrak{A}|.$$

□

The inverse relation can be stated with the lowest SP-SC ratio.

Theorem 2. *If an optimal SCO solution is given, the WCAR $\mathfrak{h}^{ao}$ can be approximated as*

$$\mathfrak{h}^{ao} = \frac{|\bigcup \mathfrak{Z}^a|}{|\bigcup \mathfrak{Z}|} \tag{63}$$
$$\mathfrak{h}^{ao} \leqslant \frac{|\bigcup \mathfrak{Z}^a|}{\frac{1}{2} + \sqrt{2\left(|\mathfrak{Z}^a|\right) + \frac{1}{4}}} \tag{64}$$

Proof. Lemma 6 states the lowest possible SPs to SC ratio, which is solved for $|\bigcup \mathfrak{Z}|$. In addition, Lemma 3 states that

$$|\mathfrak{Z}| \geqslant |\mathfrak{Z}^a|.$$

Therefore, using $|\mathfrak{Z}^a|$ instead of $|\mathfrak{Z}|$ overestimates $\mathfrak{h}^{ao}$. □

A worst case sensor placement example is presented in Figure 17. Here, the three WPNs can be covered optimally using the sensors $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$ and their resulting VFOV intersections $\Psi_{1,2}$, $\Psi_{1,3}$ and $\Psi_{2,3}$. For the SCO approximation, choosing these three SCs would be as good as choosing the combinations $\Psi_{3,4}$, $\Psi_{1,6}$ and $\Psi_{2,5}$, whereas later results in a total of six selected sensors.

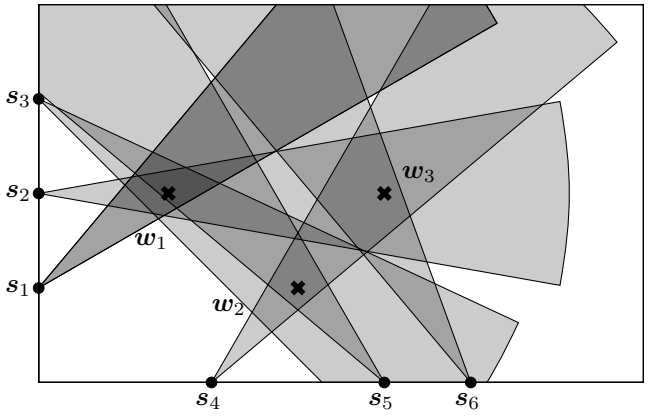


Figure 18: Six sensors $s_1, \ldots, s_6$ that cover the three WPNs w_1, w_2 and w_3 with sufficient quality. For each sensor, the VFOV is plotted in light gray. The VFOV intersections are plotted in dark gray.

5.1.2 *Iterative Solution Improvement*

The Lemmas of the WCAR derivation can be used to form heuristics that improve an approximate discrete SPPs solution. The goal is to lower the number of selected SPs by iteratively selecting an SC that includes SPs not part of any other SC and replacing the SC by combinations of already selected SPs. The result of the iterative improvement is a lower count of selected SPs, which improves an approximate SPP solution in relation to an optimal MSPQM solution.

Before introducing the search heuristics, the concept of "interchangeable combinations" has to be introduced. Interchangeable combinations are SCs of an approximate solution that can be replaced by another set of SCs to decrease the amount of SPs. An example of this is given for the SPP presented in Figure 18. If

$$\mathfrak{A} = \{\{\mathbf{s}_1, \mathbf{s}_3\}, \{\mathbf{s}_4, \mathbf{s}_5\}, \{\mathbf{s}_2, \mathbf{s}_6\}\},$$

is an initial solution of the SCO that covers the WPN w_1, w_2 and w_3. By interchanging

$$C_1 = \{\mathbf{s}_1, \mathbf{s}_3\},$$

which covers w_1, with the combination

$$\{\mathbf{s}_2, \mathbf{s}_5\},$$

the two SPs $\mathbf{s}_1$, $\mathbf{s}_3$ can be removed from the solution. In addition, the SC

$$C_1 = \{\mathbf{s}_2, \mathbf{s}_6\},$$

which covers w_3, can be replaced by

$$\{\mathbf{s}_2, \mathbf{s}_4\}$$

Data: An approximate solution $\mathfrak{A}$.
Result: The improved approximate solution $\mathfrak{A}$.

1 $\mathfrak{C}^t \leftarrow \{C \mid (C \in \mathfrak{A}) \wedge (\emptyset = C \cap (\bigcup(\mathfrak{A} \setminus C)))\}$
2 $\mathfrak{C}^s \leftarrow \{C \mid (C \in \mathfrak{A}) \wedge (\exists \mathbf{s}[(\mathbf{s} \in C) \wedge (\mathbf{s} \notin \bigcup(\mathfrak{A} \setminus C))])\}$
3 $\mathfrak{C}^i \leftarrow \mathfrak{C}^t \cup \mathfrak{C}^s$
4 $\omega(C) \coloneqq \{w_j \mid q(C, w_j) \geqslant q\}$
5 **for** $C_i \in \mathfrak{C}^i$ **do**
6 $\quad \mathfrak{C}^r \leftarrow \{\{\mathbf{s}_h, \mathbf{s}_i\} \mid (\mathbf{s}_h \in (\bigcup \mathfrak{A}) \setminus C_i) \wedge (\mathbf{s}_i \in (\bigcup \mathfrak{A}) \setminus C_i) \wedge (\mathbf{s}_h \neq \mathbf{s}_i)\}$
7 $\quad \mathfrak{W}^r \leftarrow \{\omega(C) \mid C \in \mathfrak{C}^r\}$
8 $\quad$ **if** $W \subseteq \bigcup \mathfrak{W}^r$ **then**
9 $\quad\quad \mathfrak{A} \leftarrow \mathfrak{C}^r$
10 $\quad$ **end**
11 **end**

Algorithm 5: Interchangeable Combination Replacement

to remove the SP $\mathbf{s}_6$ from the solution. In combination, the initial solution with six sensors can be iteratively improved by interchanging combinations to a solution that only contains three sensors.

To define interchangeable combinations, let $C_i \in \mathfrak{A}$ be an SC from an approximate solution to a discrete SPP. In addition, let W_i be the WPNs C_i covers. Furthermore, let

$$\mathfrak{C}^r = \{\{\mathbf{s}_h, \mathbf{s}_i\} \mid (\mathbf{s}_h \in \mathfrak{A}) \wedge (\mathbf{s}_i \in \mathfrak{A}) \wedge (\mathbf{s}_h \neq \mathbf{s}_i)\}$$

be a family of sets of SC from $\mathfrak{A}$ that covers the WPN W_r. If, and only if

$$(W_i \subseteq W_r) \wedge (|\bigcup \mathfrak{A} \setminus C_i| \leqslant |\bigcup \mathfrak{A}|)$$

$\mathfrak{C}^r$ is an interchangeable family of sets of SCs that improves the solution quality because the resulting solutions contains less SPs.

To improve the solution quality of an approximate solution, an algorithm to remove SCs that have one or two unique SPs is presented as Algorithm 5. "Unique" SP refers to the SP that is only part of one SC $C \in \mathfrak{A}$.

At first, the algorithm extracts all SCs with two unique SPs and stores them in the family of sets $\mathfrak{C}^t$ (Line 1). These are all SCs whose intersection with a flattened family of sets that contains all SPs of $\mathfrak{A}$ is the empty set, which is expressed by the condition

$$\emptyset = C \cap (\bigcup(\mathfrak{A} \setminus C)).$$

Next, all SCs are extracted and stored in the family of sets $\mathfrak{C}^s$ that have one SP unique to this SC (Line 2), which is expressed by the condition

$$\exists \mathbf{s}[(\mathbf{s} \in C) \wedge (\mathbf{s} \notin \bigcup(\mathfrak{A} \setminus C))].$$

Here, instead of both SPs of an SC not allowed to be part of any other SC, there has to exist only one SP $\mathbf{s}$ in the SC such that it is not in any other SC. All extracted combinations are combined in the family of sets $\mathfrak{C}^i$ (Line 3). Further, the function $\omega(C)$ is defined to allow the computation of visibilities between a SC and all WPNs (Line 4).

In the following loop (Line 5-10) for every SC $C_i \in \mathfrak{C}^i$ it is tested, whether the SC can be replaced by interchangeable SCs. Therefore, for every C_i all possible SCs with the SPs of $\mathfrak{A}$ and without the SPs of C_i are created and assigned to $\mathfrak{C}^r$ (Line 6). To check if $\mathfrak{C}^r$ still holds a valid solution to the SPP, the WPNs that are covered by each SC are calculated and stored in $\mathfrak{W}^r$ (Line 7). If $\mathfrak{C}^r$ still holds a valid solution (Line 8), it becomes the new approximate solution to the SPP.

The time complexity of the heuristic is mostly influenced by the update of the remaining WPN (Line 7). Here, $\omega(C)$ is executed at most

$$\frac{1}{2}\left(|\bigcup\mathfrak{A}|^2 - |\bigcup\mathfrak{A}|\right)$$

times, which is the number of unique combinations of all SPs in $\bigcup\mathfrak{A}$.

The function $\omega(C)$ itself has a linear complexity, which is dependent on the number of WPNs $|W|$. In addition, the loop is executed at most $|\mathfrak{A}|$ times, because the approximate solution can have at most $|\mathfrak{A}|$ unique SC. Altogether, the time complexity for the heuristic is

$$\mathcal{O}(|W||\mathfrak{A}|^3). \tag{65}$$

5.2 GREEDY ALGORITHMS

The SCO provides a simplified way to find a valid solution for an SPP. Nevertheless, the procedure is also based on solving a BIP model, which is an expensive procedure, especially for large input sizes. In addition, the SCO only uses indirect information on the overall SPP goal of this thesis, to reduce the number of SPs.

To provide methods that are able to solve SPP with polynomial time complexity, three heuristics have been developed. They are based on greedy approximation strategies, which themselves have a complexity that grows linear with the problem size. In addition, two of the heuristics directly select SPs and therefore are more likely to promote solutions with a smaller number of SPs.

5.2.1 *Sensor Combination Selection*

A trivial transition from the SCO to a greedy algorithm is to use a greedy "best-in" strategy (Korte & Vygen, 2010) on the SCs and their quality. The task is to choose from the SCs in $\mathfrak{C}$ until all WPNs are covered with a sufficient quality, a classic set covering problem.

Following Hromkovič (Hromkovič, 2004), a best-in strategy provides the best possible approximation ratio for set covering problems.

The algorithm—shown in Algorithm 6—iteratively chooses the SCs from $\mathfrak{C}$ that covers the maximum number of WPNs and is not already part of the solution (Line 4). Thereby, the number of covered WPN is determined by calling function $\omega(C, W)$, which calculates all WPNs of W that are covered by the SC given as C (Line 2).

A chosen SC is added to the family of sets $\mathfrak{Z}^a$ (Line 5) that contains the approximate solution when all WPNs are covered. To update the number of covered WPNs all combinations of all SPs part of the approximate solution have to be considered. Therefore, in Line 6

$$\{\omega(\{\mathbf{s}_h, \mathbf{s}_i\}, W) \mid (\mathbf{s}_h \in \bigcup \mathfrak{Z}^a) \wedge (\mathbf{s}_i \in \bigcup \mathfrak{Z}^a) \wedge (\mathbf{s}_h \neq \mathbf{s}_i)\}$$

is the family of sets that holds the set of WPNs for every possible combination of two sensors $\mathbf{s}_h$, $\mathbf{s}_i$ from the flattened family of sets $\bigcup \mathfrak{Z}^a$.

The time complexity of the heuristic is mostly influenced by the update of the remaining WPNs. Here, $\omega(C, W)$ is executed

$$\frac{1}{2}\left(|\bigcup \mathfrak{Z}^a|^2 - |\bigcup \mathfrak{Z}^a|\right)$$

times, which is the number of unique combinations of all SPs in $\bigcup \mathfrak{Z}^a$. Since $|\bigcup \mathfrak{Z}^a|$ grows in the process of solving the SPP, its upper bound, the number of all sensors

$$|\bigcup \mathfrak{C}| = |S|,$$

has to be used to approximate the time complexity on the input size.

The function $\omega(C, W)$ itself has a linear complexity, which is dependent on the size of the input set of WPNs $|W|$. In addition, the loop is executed at most $|W|$ times, because in every iteration at least one WPN is covered by a newly chosen SC.[1] Altogether, the time complexity for the heuristic is

$$\mathcal{O}(|W|^2|S|^2). \tag{66}$$

The heuristic ensures that every WPN is covered by an SC with a sufficient quality. Following Hromkovič (Hromkovič, 2004), its worst-case approximation ratio is $\log(|\mathfrak{C}|) + 1$. Thus, using the derived WCAR of the SCO (Equation 63) the WCAR of the greedy SCO is

$$\log(h^{ao}) + 1. \tag{67}$$

5.2.2 *Combined Coverage Algorithm*

An extension of the greedy SCO is the GCS, which allows not only SCs to be selected but also single SP. Therefore, it improves the probability of choosing an SP that increases the coverage with the already

1 Otherwise the model is not solvable.

Data: An SPP with WPNs W to be covered by choosing SCs C from $\mathfrak{C}$.
Result: A valid approximate solution $\mathfrak{A}$

```
1 𝔄 ← ∅
2 ω(C, W) ≔ {w_j | (w_j ∈ W) ∧ (q(C, w_j) ⩾ q)}
3 while W ≠ ∅ do
4 |   C ← argmax_{∀C∈𝔠\𝔄} [|ω(C, W)|],
5 |   𝔄 ← 𝔄 ∪ C
6 |   W ← W \ ⋃{ω({s_h, s_i}, W) | (s_h ∈ ⋃𝔄) ∧ (s_h ∈
  |    ⋃𝔄) ∧ (s_h ≠ s_i)}
7 end
```

Algorithm 6: Greedy Sensor Combination Selection

selected SPs instead of always adding an SC. Furthermore, even if an SC is chosen by the algorithm it is ensured that the SC provides the maximum increase in covered WPNs with all of the already selected SPs.

Either the algorithm chooses the SC or SP that improves the solution the most. The heuristic—shown in Algorithm 7—is based on the same basic steps as the greedy SCO algorithm with an additional distinction to select an SC or SP. The algorithm is based on the assumption that WPNs, which are already seen by an SP of the selected SCs can be covered cheaper by using only one additional SP. It first defines a function $\omega(S^r, W^r)$ that returns a set of covered WPNs from the given WPNs W^r (Line 2). To achieve this, for every SC $\{\mathbf{s}_h, \mathbf{s}_i\}$ that is possible to build with SPs in S^r, all covered WPNs are calculated

$$\{\boldsymbol{w}_j \mid (\boldsymbol{w}_j \in W^r) \wedge (q(\{\mathbf{s}_h, \mathbf{s}_i\}, \boldsymbol{w}_j) \geqslant q)\}.$$

The covered sets of WPNs are part of a family of sets, which is flattened so that only the set of covered WPNs is returned.

At first, the function is used by the algorithm to find the SP $\mathbf{s}_i \in S$ that leads to the largest increase in covered WPNs with an SP $\mathbf{s}_h \in A$ that is already part of the approximate solution (Line 4). Then the SC that covers the most WPNs is calculated from all SPs $\mathbf{s} \in S$, which are not yet part of the solution (Line 5). The number of covered WPNs for the best SP and SC are compared and either the SP or the SC is added to the approximate solution (lines 6 to 9). Finally, the set uncovered WPNs W is reduced by removing all WPNs covered by the set of selected SPs A and the set of SPs S is reduced by removing all selected SPs from A.

The time complexity of the heuristic is determined by the function $\omega(S^r, W^r)$, which is called to determine the next best SP or SC and to update A. The two calls in the comparisons (Line 6) can be neglected because the values might also be saved in the prior search. To find

Data: The set of SPs S and the set of WPNs W of an SPP.
Result: The set of SPs A that forms valid approximate solution.

$C \leftarrow \emptyset$
$\omega(S^r, W^r) \coloneqq \bigcup\{\{\boldsymbol{w}_j \mid (\boldsymbol{w}_j \in W^r) \wedge (q(\{\mathbf{s}_h, \mathbf{s}_i\}, \boldsymbol{w}_j) \geqslant q)\} \mid (\mathbf{s}_h \in S^r) \wedge (\mathbf{s}_i \in S^r) \wedge (\mathbf{s}_h \neq \mathbf{s}_i)\}$
while $W \neq \emptyset$ **do**
  $\mathbf{s}_{max} \leftarrow \operatorname{argmax}_{\forall \mathbf{s}_i \in S} [|\omega(A \cup \mathbf{s}_i, W|]$
  $C_{max} \leftarrow \operatorname{argmax}_{\forall \{\mathbf{s}_h, \mathbf{s}_i\}, (\mathbf{s}_h \neq \mathbf{s}_i) \wedge (\mathbf{s}_h \in S) \wedge (\mathbf{s}_i \in S)} [|\omega(A \cup \{\mathbf{s}_h, \mathbf{s}_i\}, W)|]$
  if $|\omega(A \cup \mathbf{s}_{max}, W)| \geqslant |\omega(C^{max}, W)|$ **then**
    $A \leftarrow A \cup \mathbf{s}_{max}$
  else
    $A \leftarrow A \cup C_{max}$
  end
  $W \leftarrow W \setminus \omega(A, W)$
  $S \leftarrow S \setminus A$
end

Algorithm 7: Greedy Combined Selection

the next best SP, the $\omega(S^r, W^r)$ is called $|S|$ times and to find the next best SC, the function is called

$$\frac{1}{2}\left(|S|^2 - |S|\right) \tag{68}$$

times, which is the number of unique combinations of all SPs in S. Each function call has a time complexity of

$|W^r|$ to check every WPN for coverage $(q(\{\mathbf{s}_h, \mathbf{s}_i\}, \boldsymbol{w}_j) \geqslant q)$,

$\frac{1}{2}(|S^r|^2 - |S^r|)$ to perform the check for every unique SC possible,

$|W^r|\frac{1}{2}(|S^r|^2 - |S^r|)$ to flatten the final family of sets.

Altogether, the time complexity of the function $\omega(S^r, W^r)$ can be given by multiplying the first two dependent operations and adding the independent complexity of the flattening as

$$|W^r|\frac{1}{2}(|S^r|^2 - |S^r|) + |W^r|\frac{1}{2}(|S^r|^2 - |S^r|) = |W^r|(|S^r|^2 - |S^r|).$$

Considering that the loop runs at maximum $|W|$ times until all WPNs are covered, this is also the maximum number of times the best SP and SC is to be calculated. The overall time complexity can be stated by considering the number of runs, function calls per run and the time complexity of the function as

$$|W|(\frac{1}{2}(|S|^2 - |S|) + |S|)|W^r|(|S^r|^2 - |S^r|).$$

Using S and W as an upper bound of S^r and W^r, the complexity can be stated as

$$\mathcal{O}(|W|^2|S|^4). \tag{69}$$

Altogether, the GCS provides a method of choosing the next SP or SC that considers the increase in combination with all already selected SPs on the cost of an increased time complexity.

5.2.3 *Single Sensor Selection*

In the last presented greedy algorithm, the GSSS, the concepts of the GCS are further refined. The idea is to only allow a single SP to be added in each iteration, by making use of a pre-calculated single sensor coverage of every WPN. The goal of this method is to increase the probability of large VFOV intersections that cover many WPNs. By not considering the SCs in the initial 1-coverage, SPs with a high number of covered WPNs are chosen. To return to the example stated in Figure 18, an initial single coverage that minimized the number of selected SPs contains for instance the sensors $\mathbf{s}_2$ and $\mathbf{s}_3$. With these two SPs selected, the SP that increases the number of covered SPs the most is $\mathbf{s}_4$. Selecting $\mathbf{s}_4$ leads to an optimal solution of the discrete SPP.

To find an initial single sensor cover, a BIP model similar to the SCO can be applied that exploits the VF. Let $\mathbf{x}^s$ be the binary decision vector for SPs (Equation 26)

$$\text{minimize} \quad \sum_{\forall i} x_i^s \tag{70}$$

$$\text{subject to} \quad \sum_{\forall i} x_i^s v(i,j) \geqslant 1, \quad \forall\, j \tag{71}$$

An optimization solution can be transformed to a set of selected sensors as

$$Z = \{\mathbf{s}_i \mid (\mathbf{s}_i \in S) \wedge (x_i^s = 1)\}.$$

The GSSS assumes Z to be a set of SPs that form a minimum 1-cover of all WPNs.

The GSSS, shown in Algorithm 8, is based on removing items from a set W that holds all WPNs, which are not covered with a sufficient quality (Line 2). As the GCS, it relies on a function $\omega(S^r, W^r)$ that calculates all covered WPNs of a set W^r that can be covered with any unique combination of SPs out of a given set S^r. Iteratively, the best SP that covers the most WPNs with the already selected SPs is calculated in Line 4. The found SPs is added to the approximate solution A.

Data: An initial single coverage solution Z^s.
The set of SPs S and the set of WPNs W of an SPP.
Result: The set of SPs A that forms valid approximate solution.

1 $\omega(S^r, W^r) \coloneqq \bigcup\{\{\boldsymbol{w}_j \mid (\boldsymbol{w}_j \in W^r) \wedge (q(\{\boldsymbol{s}_h, \boldsymbol{s}_i\}, \boldsymbol{w}_j) \geqslant q)\} \mid (\boldsymbol{s}_h \in S^r) \wedge (\boldsymbol{s}_i \in S^r) \wedge (\boldsymbol{s}_h \neq \boldsymbol{s}_i)\}$
2 $W \leftarrow W \setminus \omega(Z^s, W)$
3 **while** $W \neq \emptyset$ **do**
4 $\quad \boldsymbol{s}_{max} \leftarrow \operatorname{argmax}_{\forall \boldsymbol{s}_i \in S} [|\omega(A \cup \boldsymbol{s}_i)|]$
5 $\quad$ **if** $|\omega(\boldsymbol{s}_{max})| \geqslant 0$ **then**
6 $\quad\quad A \leftarrow A \cup \boldsymbol{s}_{max}$
7 $\quad$ **else**
8 $\quad\quad C_{max} \leftarrow \operatorname{argmax}_{\forall\{\boldsymbol{s}_h, \boldsymbol{s}_i\}, (\boldsymbol{s}_h \neq \boldsymbol{s}_i) \wedge (\boldsymbol{s}_h \in S) \wedge (\boldsymbol{s}_i \in S)} [|\omega(A \cup \{\boldsymbol{s}_h, \boldsymbol{s}_i\}, W)|]$
9 $\quad\quad A \leftarrow A \cup C_{max}$
10 $\quad$ **end**
11 $\quad W \leftarrow W \setminus \omega(A, W)$
12 $\quad S \leftarrow S \setminus A$
13 **end**

Algorithm 8: Greedy Single Sensor Selection

A special case may occur if there is no SP that forms an SC with the selected SPs that leads to a decrease in WPNs. This might occur because the initial selection does not necessary contain an SP for every WPN, which can form an SC that covers the WPNs with a sufficient quality. In this case, the algorithm selects the best SC as can be seen in lines 8 to 9. Finally, the sets of uncovered WPNs and remaining SPs are updated (Line 11 and 12).

The time complexity can be derived analogue to the GCS. The difference is that a search for the SC that covers the most WPNs with the selected SPs is usually not performed. Therefore, it is neglected in the time complexity calculation. Thus, in comparison to the time complexity of the GCS, Equation 68 must be changed to include only the search for the best SP (Line 4). This takes only $|S|$ calls of $\omega(S^r, W^r)$ instead of $\frac{1}{2}(|S|^2 - |S|) + |S|$. Therefore, the resulting time complexity for the GSSS is the time complexity of the GCS divided by $|S|$,

$$\mathcal{O}(|W|^2|S|^3). \tag{72}$$

The reduced complexity is another advantage of the GSSS over the GCS.

5.3 DIVIDE AND CONQUER

An alternative approach to find a valid solution in reasonable time is a separation of the problem into small sub problems, which can be solved efficiently to optimality. If possible, the sub problems are to be solved in parallel; an additional way to save processing time. The quality of a joint solution of all sub problem solutions depends on how well the initial problem is separated. In the best case, the joint solution is an optimal solution to the initial problem. Hence, an optimal division results in non-overlapping sub problems.

The goal of applying this approach to the SPP is to exploit the solution quality of solving the MSPQM to optimality even if the initial SPP does not allow such a computation in reasonable time. Since solving the MSPQM gives an optimal solution to the discrete SPP, the quality of a combination of solutions for parts of the problem is determined by the quality of the separation. The presented approach to partition an SPP is based on exploiting the geometric properties of its environment.

An optimal separation of an environment is a set of polygons that are separated by walls, mountable objects or obstacles. This contradicts the environment definition (see Section 2.6.1) and therefore the goal of the presented approach is to partition the environment polygon in a way that minimizes the visibilities among the partitioned polygons.

The important aspects to achieve this are:

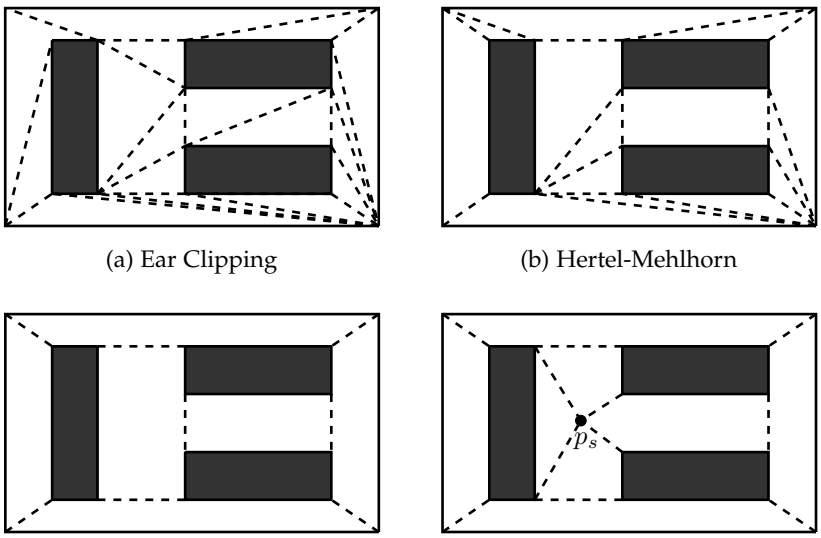
VISIBILITY Each sub polygon should have as few sensors as possible whose VFOV spans over to neighboring sub polygons.

SOLVABILITY The sensor positioning must be independently solvable for each sub polygon.

SIZE The sensor positioning in all sub polygons should be solvable in reasonable time.

The idea is to use a 2d convex polygon decomposition to split the simple 2d polygon of the environment into convex parts. Since the definition of a convex sub polygon implies that every point within is visible from every other point, two convex sub polygons of a decomposition are usually separated by a reflex edge, which blocks some of the sensor VFOVs. Thus, by dividing the polygon at the reflex vertices it is tried to minimize the visibilities among the sub polygons.

In general, two classes of convex decomposition algorithms exist, one that use additional vertices and one that do not (Goodman & O'Rourke, 2004). The first class may introduce new vertices anywhere within the polygon, the so-called Steiner points. The second class only operates on the vertices of the polygon and introduces new edges between them to split the polygon. According to Chazelle and Palios



(a) Ear Clipping (b) Hertel-Mehlhorn

(c) Minimum Polygons (d) Steiner Point

Figure 19: Four different decompositions of a polygon with holes. The first one 19a uses the ear-clipping algorithm, the second one 19b improves the triangulation solution with the Hertel-Mehldorn heuristic and the third one 19c was decomposed manually, to show a minimum decomposition without Steiner points with respect to the number of convex polygons. The last one 19d was also added manually to show a decomposition including a Steiner point ($\mathbf{p}_s$).

(Chazelle & Palios, 1994) a convex polygon decomposition into a minimum number of convex pieces without using Steiner points can be calculated in $\mathcal{O}(n^3)$ for polygons without holes. For polygons with holes, the problem becomes NP-hard (Keil, 2000) but can be approximated in $\mathcal{O}(n \log n)$.

The quality of convex polygon decomposition algorithms are typically measured with respect to an optimal decomposition with the least possible number of convex polygons. For a separation of the SPP, the resulting number of convex polygons is not important. In contrast, a decomposition that produces well-separated small convex polygons is to be preferred because the runtime of solving the divided SPP will be determined by the largest convex polygon.

To give an example of commonly used polygon decomposition algorithms that do not use Steiner points, Figure 19 shows an example of the ear-clipping triangulation method (ElGindy, Everett, & Toussaint, 1993) and the Hertel-Mehlhorn heuristic (Hertel & Mehlhorn, 1985) that can be used to improve the triangulation to create a smaller number of convex polygons. The disadvantage of these algorithms is their creation of diagonal cuts. These lead to several small convex polygons, which are not separated from their neighboring polygons with reflex edges. Thus, these pieces are more likely to lead to a se-

lection of SCs that also covers WPNs in the neighboring convex polygons.

In contrast to the optimal convex decomposition without Steiner points, a convex decomposition using Steiner points might introduce points within the environment, as shown in Figure 19d. At these points, all adjoining polygons are also not well separated from each other. Since the defined SPP does not allow sensors to be placed inside the polygons, allowing Steiner points everywhere in the environment may lead to situations as presented in the center of the surrounding polygon. Here, a Steiner point is the connection vertex for four convex polygons. This leads to WPNs in near proximity to the Steiner point that are likely be covered by SCs from each of the four convex parts.

To provide a trade-off that makes use of the positive aspect of Steiner points while it tries to avoid the creation of small triangular polygons, a new convex decomposition algorithm is introduced that allows Steiner points only on the boundary. With this restriction, the polygon decomposition is more likely to create convex polygons that include at least two edges of the boundary and therefore a higher probability that WPNs in them can be covered with sensors placed on these edges.

5.3.1 *Radial Convex Polygon Decomposition*

The idea of the Radial Convex Polygon Decomposition (RCPD) is that every reflex vertex of a polygon can be removed by adding an edge that runs radial from the vertex until it hits an edge of the polygon. These edges will be called radial edges. By inserting a radial edge at every reflex vertex, the polygon is automatically separated into convex pieces. However, the separation might introduce Steiner points within the polygon, as shown in Figure 19d. Therefore, in a second step, the algorithm removes the Steiner points by rearranging the radial edges.

The algorithm itself has three sub-algorithms—in the following called procedures—the Steiner point removal, the reflex vertex check and the convex polygon creation. They are presented in Algorithm 9, 10 and 11.

The first procedure finds and removes all Steiner points within the environment given a set of reflex edges. Figure 20 shows reflex edges as dotted lines. Two of them originate at the reflex vertices $\mathbf{p}_x$ and $\mathbf{p}_y$ and extend to $\mathbf{p}_{x'}$ and $\mathbf{p}_{y'}$ (see Figure 20). The length of the edges is limited by the first intersection between the ray (Equation 1) that originates at $\mathbf{p}_x$ and $\mathbf{p}_y$ and the polygon P. With the radial edges, the Steiner points within the environment are calculated by performing an intersection of any tuple of radial edges (Line 1).

Data: A polygon P with edges E^p and radial edges E^r
Result: A cleaned set of radial edges E^r

$\mathfrak{P}^s = \{\{\boldsymbol{e}_i, \boldsymbol{e}_h\} \mid (\boldsymbol{e}_i \in E^r) \wedge (\boldsymbol{e}_h \in E^r) \wedge (\boldsymbol{e}_h \neq \boldsymbol{e}_i) \wedge (\emptyset \neq \boldsymbol{e}_i \sqcap \boldsymbol{e}_h)\}$
$E^m \leftarrow \emptyset$
while $\mathfrak{P}^s \neq \emptyset$ **do**
  $\mathfrak{P}^m \leftarrow \{\{\boldsymbol{e}_m, \boldsymbol{e}_r\} \mid (\boldsymbol{e}_m \in E^m) \wedge (\boldsymbol{e}_r \in E^r) \wedge (\emptyset \neq \boldsymbol{e}_m \sqcap \boldsymbol{e}_r)\}$
  if $\mathfrak{P}^m = \emptyset$ **then**
    $\boldsymbol{e}_s \coloneqq (\boldsymbol{p}_s, \boldsymbol{p}_x), \qquad \boldsymbol{e}_t \coloneqq (\boldsymbol{p}_t, \boldsymbol{p}_y)$
    $P_s \leftarrow \operatorname{argmin}_{\forall\{\boldsymbol{e}_s, \boldsymbol{e}_t\} \in \mathfrak{P}^s, \emptyset = (\boldsymbol{p}_s, \boldsymbol{p}_t) \sqcap P} [\|\boldsymbol{p}_s, \boldsymbol{p}_t\|]$
    $E^m \leftarrow E^m \cup (\boldsymbol{p}_s, \boldsymbol{p}_t)$
    $E^r \leftarrow E^r \setminus \{\boldsymbol{e}_s, \boldsymbol{e}_t\}$
    $E^r \leftarrow E^r \cup$ {`Reflex vertex check`$(\boldsymbol{p}_t, \boldsymbol{p}_s)$,`Reflex vertex check`$(\boldsymbol{p}_s, \boldsymbol{p}_t)$}
  else
    $\{\boldsymbol{e}_m, \boldsymbol{e}_r\} \coloneqq P_m \in \mathfrak{P}^m$
    $\boldsymbol{e}_m \coloneqq (\boldsymbol{p}_m, \boldsymbol{p}_n), \qquad \boldsymbol{e}_r \coloneqq (\boldsymbol{p}_r, \boldsymbol{p}_y)$
    if $\|\boldsymbol{p}_m, \boldsymbol{p}_r\| \leqslant \|\boldsymbol{p}_n, \boldsymbol{p}_r\|$ **then**
      $\boldsymbol{e}_m \leftarrow (\boldsymbol{p}_r, \boldsymbol{p}_m)$
    else
      $\boldsymbol{e}_m \leftarrow (\boldsymbol{p}_r, \boldsymbol{p}_n)$
    end
    $\boldsymbol{p}_e \leftarrow \boldsymbol{e}_m \sqcap P$
    if $\boldsymbol{p}_e \neq \emptyset$ **then**
      $\boldsymbol{e}_m \leftarrow (\boldsymbol{p}_r, \boldsymbol{p}_e)$
    end
    $E^m \leftarrow E^m \cup \boldsymbol{e}_m$
    $E^r \leftarrow E^r \setminus \boldsymbol{e}_r$
    $E^r \leftarrow E^r \cup$ {`Reflex vertex check`$(\boldsymbol{p}_m, \boldsymbol{p}_r)$}
  end
  $\mathfrak{P}^s = \{\{\boldsymbol{e}_i, \boldsymbol{e}_h\} \mid (\boldsymbol{e}_i \in E^r) \wedge (\boldsymbol{e}_h \in E^r) \wedge (\boldsymbol{e}_h \neq \boldsymbol{e}_i) \wedge (\boldsymbol{e}_i \sqcap \boldsymbol{e}_h)\}$
end

Algorithm 9: Steiner Point Removal

Data: A polygon P and two points on its rings $\boldsymbol{p}_t, \boldsymbol{p}_s$.
Result: A radial edge $\boldsymbol{p}_r$.

$\gamma^{max} \leftarrow \max\{\gamma(\boldsymbol{p}_t, \boldsymbol{p}_s, \boldsymbol{p}_{s-1}), \gamma(\boldsymbol{p}_t, \boldsymbol{p}_s, \boldsymbol{p}_{s+1})\}$
if $\gamma^{max} > 180°$ **then**
  $\vartheta \leftarrow (\boldsymbol{p}_s, \gamma^{max})$
  $\boldsymbol{p}_x \leftarrow \min_{c \in \vartheta \sqcap P} [\|c, \boldsymbol{p}_s\|]$
  $\boldsymbol{p}_r \leftarrow (\boldsymbol{p}_s, \boldsymbol{p}_x)$
end

Algorithm 10: Reflex Vertex Check

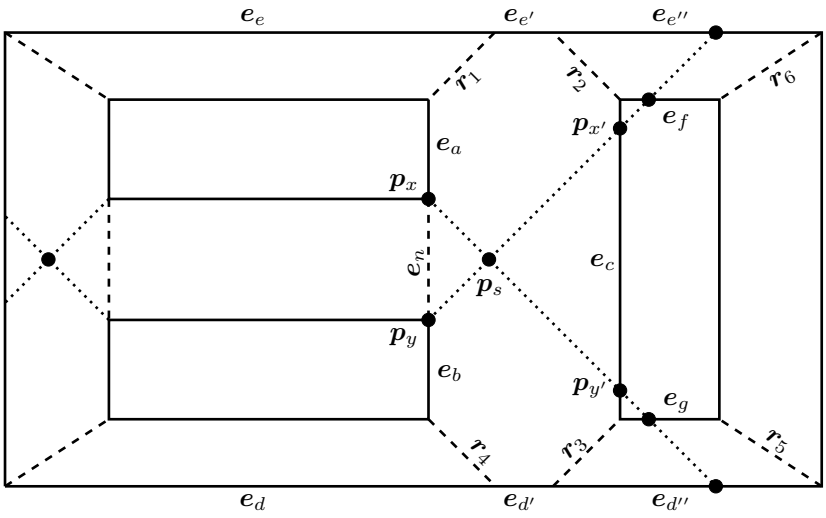


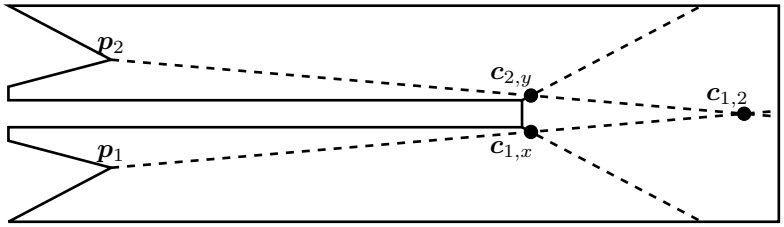
Figure 20: Polygon with properties that are used in the RCPD. The reflex vertices p_x and p_y are to be removed by the RCPD algorithm. The polygon intersections $p_{x'}$ and $p_{y'}$ represent the end of the respective radial edge. Since their intersection creates the Steiner point p_s, they are replaced by the newly introduced edge e_n. The remaining radial edges are labeled with r_i, whereas the polygon edges are labeled from e_a to e_g.

In the following loop (Line 3 to 27), the Steiner points are subsequently removed. For this, the set of merged radial edges E^m is used. Over all sets of radial edges $\{e_s, e_t\} \in \mathfrak{P}^s$ that intersect and form a Steiner point, the one with the shortest distance of their originating reflex vertices is selected (Line 7). Their reflex vertices are connected by a new edge (p_s, p_t) that is added to the set of merged edges (Line 8). Further, the two intersecting edges are removed from the set of reflex edges E^r (Line 9). For both points of the new edge it is checked if they are still reflex and if so, their reflex edges are added to E^r (Line 10).

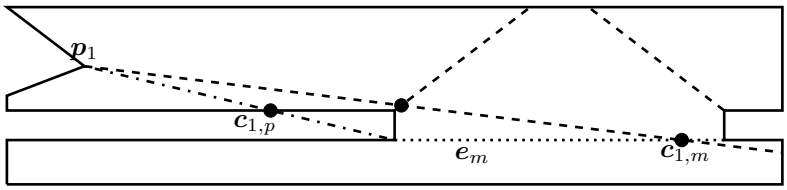
A condition of the selection in line 7 is $\emptyset = (p_s, p_t) \sqcap P$, which handles the special case where the redirection leads to an intersection with the polygon. An example of this is shown in Figure 21a. Here, $c_{1,2}$ is the Steiner point that would be removed first, which is not possible because an edge connecting p_1 and p_2 would be outside the polygon boundary. Therefore, this intersection is ignored and one of the Steiner points $c_{1,x}$ or $c_{2,y}$ is used because, a merged edge that intersects with the polygon boundary can only occur if other Steiner Points have not been merged.

This is true, because the radial edges e_r and e_s intersect somewhere in the polygon and the edge that connects both of their reflex vertices p_r and p_s intersects the polygon boundary. Therefore, it must form an inward spike, as shown in Figure 21a. Such a spike must have at least one reflex vertex with a radial edge that lies between p_r and p_s. Ergo, it intersects at least one of the radial edges e_r or e_s.

The handled Steiner point is then removed through removal of the two causing reflex edges from E^r (Line 9) and it is ensured that none



(a) Intersection Merged/Polygon Edge



(b) Intersection Radial/Polygon Edge

Figure 21: Special cases that occur in RCPD. In 21a an intersection of two radial edges $\mathbf{c}_{1,2}$ is shown that would be chosen next because the distance of the two vertices $\mathbf{p}_1$ and $\mathbf{p}_2$ are the smallest distance of all intersecting radial edges. These two vertices cannot be connected by a merged edge because this edge would intersect the polygon boundary.
In 21b the dotted line $\boldsymbol{e}_m$ is a merged edge that intersects with the dashed radial edge of $\mathbf{p}_1$ at $\mathbf{c}_{1,m}$. Therefore, it will be connected to the nearest reflex vertex of $\boldsymbol{e}_m$. This connection is drawn with a dashed dotted line that intersects the polygon boundary at $\mathbf{c}_{1,p}$.

of the two formally reflex vertices got reflex again due to the shift in the radial edge direction (Line 10). Finally, the existing Steiner points are updated (Line 27).

To check for reflex vertices of merged edges, the procedure "reflex vertex check" (Algorithm 10) computes if the vertex $\mathbf{p}_s$ is still reflex. This is done checking the inner angle between the newly created edge $(\mathbf{p}_t, \mathbf{p}_s)$ and the two edges that $\mathbf{p}_s$ connects (Line 1). If any of them is still reflex, the algorithm computes the nearest intersection point of an edge dividing ray (Line 3) with the polygon (Line 4). Finally, it returns the newly created radial edge.

An example of the removal procedure can be seen in Figure 20, where the Steiner point P_s is removed by replacing the two radial edges at $\mathbf{p}_s$, $\mathbf{p}_t$ with a new edge $\boldsymbol{e}_n$ that connects the two reflex vertices. Merged edges serve as anchors, therefore it is tried to merge radial edges first that originate in polygon vertices which are close to each other.

After the first reflex edge has been merged, there might also be merged edges that intersect with radial edges. These Steiner points are calculated as a family of sets $\mathfrak{P}^m$ that contains the two intersecting edges for every Steiner point (Line 4). If such intersections exist (Line 5), one of them is chosen (Line 12) and the radial edge that intersects with the merged edge is moved to the nearest of their vertices

(lines 14 to 17). A redirection may result in an intersection with the polygon boundary, which is tested in Line 20. In case an intersection exist, the edge is shortened to its intersection point with the boundary, as exemplary shown in Figure 21b. Because a Steiner point of a radial merged edge intersection is prioritized over radial edge intersections, in the example $\mathbf{c}_{1,m}$ is chosen and handled by moving the radial edge of $\mathbf{p}_1$ to the nearest point of $\boldsymbol{e}_m$. The resulting edge intersects the polygon boundary at $\mathbf{c}_{1,p}$ and is simply shortened to end here.

Finally, the moved reflex edge is added to the set of merged edges (Line 23) and removed from the set of reflex edges (Line 24). Further, the reflex vertex is checked whether it got reflex again (Line 25).

In conclusion, the Steiner point removal algorithm handles the intersections of radial edges by merging and the intersections of radial edges with merged edges by redirection. Thus, after the algorithm finishes, no more Steiner points within the polygon boundary exist.

In the next step, the convex polygon creation algorithm extracts the introduced polygons. The idea of the algorithm is that every convex polygon can be extracted by starting at an intersecting edge ($e \in E^p \vee E^r$) and following the connecting edges with the smallest inner angle until the convex polygon is closed. To prepare this, every reflex edge is added in opposite direction (Line 2). Furthermore, the additional vertices on the boundary are added to the polygon (Line 3). In the second loop (Line 5), one of the polygon edges is chosen as a start point. Then, the inner loop always adds the next connecting edge with the smallest inner bearing angle (Line 10) to extract the convex sub polygon or ring. The next edge that is added is removed either from the set of polygon or reflex edges (lines 12 to 16) and added to the convex sub polygon $\mathbf{r}$. After all edges have been to the sub polygon, which is indicated by the initial vertex, it is added to the multi polygon of convex polygons $\mathfrak{P}^C$ in Line 19.

In Figure 20, the inner loop at Line 9 would start exemplary at $\boldsymbol{e}_f$ and then add $\mathbf{r}_6$, $\boldsymbol{e}_{e''}$ and $\mathbf{r}_2$ to the ring r before it stops because the end point of edge $\mathbf{r}_2$ is identical with the start point of edge $\boldsymbol{e}_f$. In the extraction, the forward and backward edge of $\boldsymbol{e}_f$ and $\boldsymbol{e}_{e''}$ is removed from E^p, because they do not need to be considered anymore. The opposite direction of the edge $\mathbf{r}_2$ in contrast will be needed in the extraction process of the convex polygon defined by the edges $\mathbf{r}_2$, $\boldsymbol{e}_c$, $\mathbf{r}_3$, $\boldsymbol{e}_{d'}$, $\mathbf{r}_4$, $\boldsymbol{e}_b$, $\boldsymbol{e}_n$, $\boldsymbol{e}_a$, $\mathbf{r}_1$, $\boldsymbol{e}_{e'}$. Altogether, every radial edge will be visited in forward and backward direction in the extraction process, whereas every polygon edge will only be visited in one direction.

5.3.2 *Sensor Placement in Divided Environments*

The placement in a set of convex polygons or multi polygon $\mathfrak{P}^C$ can be done iteratively or in parallel. Both strategies require a preprocessing

Data: A polygon P with edges E^p
A list of radial edges E^r
Result: A multi polygon of convex polygons $\mathfrak{P}^C$

foreach $\boldsymbol{e}_r \leftarrow (p_r, p_s) \in E^r$ **do**
  $E^r \leftarrow E^r \cup (\mathbf{p}_s, \mathbf{p}_r)$
  $P \leftarrow P \cup \{\mathbf{p}_s, \mathbf{p}_r\}$
end
while $\boldsymbol{e}_p \leftarrow (\mathbf{p}_i, \mathbf{p}_h) \in E^p$ **do**
  $r \leftarrow \boldsymbol{e}_p$
  $E^p \leftarrow E^p \setminus \boldsymbol{e}_p$
  repeat
    $r = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n),\ \boldsymbol{e}_n = (\mathbf{p}_n, \mathbf{p}_o)$
    $\mathbf{p}_{n+1} \leftarrow \operatorname{argmin}_{\forall \mathbf{p} \in P \cup E^r}[\gamma(\mathbf{p}_n, \mathbf{p}_o, \mathbf{p})]$
    $\boldsymbol{e}_{n+1} \leftarrow (\mathbf{p}_o, \mathbf{p}_{n+1})$
    if $\boldsymbol{e}_{n+1} \in P$ **then**
      $E^p \leftarrow E^p \setminus \boldsymbol{e}_{n+1}$
    end
    else
      $E^r \leftarrow E^r \setminus \boldsymbol{e}_{n+1}$
    end
  until $\mathbf{p}_o \neq \mathbf{p}_i$
  $\mathfrak{P}^C \leftarrow \mathfrak{P}^C \cup r$
end

Algorithm 11: Convex Polygon Creation

step that extracts a local set of SPs, WPNs and SCs for every sub problem from the global problem description. The extraction can be conducted by subsequently removing polygon $P_i \in \mathfrak{P}^C$ from $\mathfrak{P}^C$ and defining its placement problem by using a subset of SPs S_i, SCs C_i and WPNs W_i from the global problem definition as

$$W_i = \{\boldsymbol{w} \mid \boldsymbol{w} \sqsubseteq P_i\}, \tag{73}$$

$$S_i = \{\mathbf{s}_i \mid (\mathbf{s}_i \in S) \wedge \exists \boldsymbol{w}\,[(\boldsymbol{w} \in W_i) \wedge (v(\mathbf{s}_i, \boldsymbol{w}) = 1)]\}, \tag{74}$$

$$\mathfrak{C}_i = \{\{\mathbf{s}_h, \mathbf{s}_i\} \mid (\mathbf{s}_h \in S_i) \wedge (\mathbf{s}_i \in S_i) \wedge (\mathbf{s}_h \neq \mathbf{s}_i)\}. \tag{75}$$

Here, the WPNs in P_i (Equation 73) provide the basis to extract the SPs (Equation 74), which in turn provide the basis to extract the SCs (Equation 75). Thus, SP $\mathbf{s}_i$ are added to S_i if there exists a $\boldsymbol{w}$ in W_i such that $\mathbf{s}_i$ is visible from $\boldsymbol{w}$.

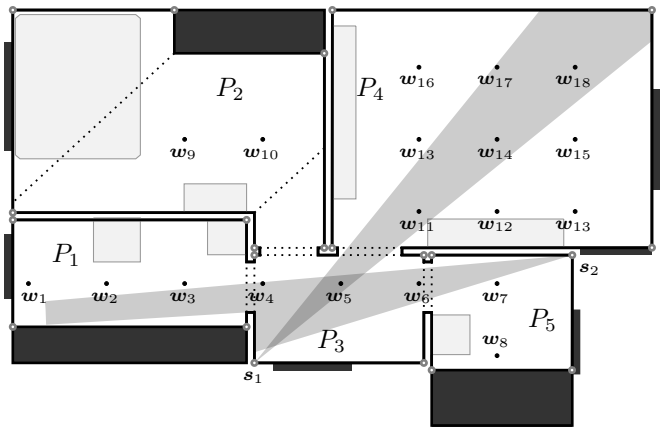


Figure 22: An environment divided into multiple convex pieces. The edges that divide the environment are shown as dotted lines. In addition, the convex polygons are named from P_1 to P_5 and the WPNs are named from w_1 to w_{18}. Finally, the initial SPs on the convex vertices of the boundary are shown as gray marks and the VFOVs of two exemplary SPs s_1 and s_2 are drawn.

An example of a separated SPP is given in Figure 22. Here, each polygon that contains WPNs, P_1 to P_5, stats a subproblem. The WPNs sets are

$$
\begin{aligned}
W_1 &= \{w_1, w_2, w_3\},\\
W_2 &= \{w_9, w_{10}\},\\
W_3 &= \{w_4, w_5, w_6\},\\
W_4 &= \{w_{11}, \ldots, w_{18}\},\\
W_5 &= \{w_7, w_8\}.
\end{aligned}
$$

The corresponding sets of SPs and SCs are omitted in the example, but with the general concept of visibility introduced in Section 3.2 and the given example of positions for SPs the problem separation should be readily understandable. In addition, it can be observed that s_1 will belong to the set of SPs S_3 and S_4 and s_2 to S_3 as well as the SC $\{s_1, s_2\}$ will be part of $\mathfrak{C}_3$.

To solve an SPP iteratively based on a subproblem separation, the sets of a subproblem W_i, S_i and the family of sets $\mathfrak{C}^i$ are used to state an MSPQM or BSPQM. The iterative placement scheme then:

1. Starts with sub problem i.
2. Calculates, a placement $Z_i \subseteq S_i$.
3. Adds Z_i to a global placement $Z = Z \cup Z_i$.
4. Moves to the next sub problem h.

However, step 2 may profit from the already calculated solutions Z because these sensors will definitively be part of the global solution. For instance, if the SP $\mathbf{s}_1$ is selected for Z_3 in the example given in Figure 22, and the placement Z_4 is calculated afterwards, $\mathbf{s}_1$ can be defined as selected in the MSPQM by forcing

$$x_1^s = 1$$

as an additional constraint in the MSPQM presented in Section 4.2.

To further improve the runtime of the polygon placement, the problems can be processed in parallel by solving every local problem independently. The global solution is a union of all sets of SPs of the local solutions. Nevertheless, to limit the scope of the experiments, only the iterative placement will be considered in the evaluations.

6 CONTINUOUS OPTIMIZATION MODELS

After possibilities to model and solve SPPs in a discrete domain have been explored, this chapter introduces possibilities to solve the problem in a continuous domain. At first, the common properties of the continuous SPP are introduced (Section 6.1). Then, two different approaches are presented to model the problem. One focuses on covering predefined WPNs with the best quality possible (Section 6.2), and one focuses on covering the whole workspace with a predefined quality without using sampled WPNs (Section 6.3). In addition, an iterative positioning scheme is presented (Section 6.4) that describes how the continuous problems can be used in combination with results from the discrete problems to further improve them.

6.1 CONTINUOUS SENSOR MODEL

All continuous problems optimize the pose of a predefined number of sensors according to the optimization goal. In contrast to the sensor state that were defined in Chapter 2.1 (Equation 5), the state of a sensor is defined in the continuous domain using all placeable edges. These are all edges that are on the walls or mountables

$$E^p = \left\{ \boldsymbol{e} \mid (\boldsymbol{e} \in \mathfrak{P}^{\text{wall}}) \vee (\boldsymbol{e} \in \mathfrak{P}^{\text{mnt}}) \right\}. \tag{76}$$

Instead of using a 2d pose that describes a position and orientation as

$$\mathbf{s} = \begin{pmatrix} s_x \\ s_y \\ s_\varphi \end{pmatrix},$$

the sensor position is restricted to the placeable edges as

$$\mathbf{s}^\rho = \begin{pmatrix} \rho \\ \varphi \end{pmatrix}.$$

Here, $0 \leqslant \rho \leqslant 1$ is a value that states the position on the placeable edges. To transform ρ into a 2d position, a lookup table of placeable edges has to be created that holds the ratio of their respective length to the overall length.

Using the set of placeable edges E^p an ordered list $\mathbf{l}$ can be created to serve as such a lookup. With the total length of all edges

$$m = \sum_{\forall \boldsymbol{e} \in E^p} \|\boldsymbol{e}\|$$

$\mathbf{l}$ can be defined to hold the ratio of each edge to the overall edge lengths as

$$\mathbf{l} = \left\{ \frac{\|\boldsymbol{e}\|}{\mathfrak{m}} \mid \boldsymbol{e} \in \mathrm{E}^{\mathrm{p}} \right\}.$$

The unique 2d position to a value ρ can be found by calculating

$$\mathfrak{i}, \qquad (\mathfrak{i} \in \mathbb{N}) \wedge (\sum_{0 \leqslant \mathfrak{i}} \mathbf{l}_{\mathfrak{i}} \leqslant \rho) \wedge (\sum_{0 \leqslant \mathfrak{i}+1} \mathbf{l}_{\mathfrak{i}} > \rho),$$

whereas $\mathfrak{i}$ is the index of the edge where the sensor is positioned. The exact position is

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_{\mathfrak{i}} \\ y_{\mathfrak{i}} \end{pmatrix} + \frac{\left(\rho - \sum_{\forall \mathfrak{j} \leqslant \mathfrak{i}} \mathbf{l}_{\mathfrak{j}}\right)}{\mathbf{l}_{\mathfrak{i}}} \begin{pmatrix} x_{\mathfrak{i}+1} - x_{\mathfrak{i}} \\ y_{\mathfrak{i}+1} - y_{\mathfrak{i}} \end{pmatrix} \leftrightarrow \boxdot(\rho),$$

which calculates the distance from the first vertex $\mathbf{p}_{\mathfrak{i}}$ of edge $\mathfrak{i}$ by subtracting the ρ-value for $\mathbf{p}_{\mathfrak{i}}$ ($\sum_{\forall \mathfrak{j} \leqslant \mathfrak{i}} \mathbf{l}_{\mathfrak{j}}$) from the given ρ-value. Dividing the result by the edge length results in a ratio that defines the distance of the unique 2d position from the first point $\mathbf{p}_{\mathfrak{i}}$ of the edge $\boldsymbol{e}_{\mathfrak{i}}$. Finally, the multiplication of the ratio with a vector that defines the edge direction and length leads to an offset from $\mathbf{p}_{\mathfrak{i}}$ that can be added to get the unique 2d position of ρ.

The notation $\boxdot(\rho)$ will be used to represent a function that does the 2d position calculation.

6.2 MAXIMUM QUALITY MODEL

The first continuous optimization model has the objective to maximize the quality at each WPN by repositioning a set of preselected SPs. The idea of the model is that by moving away from sampled SPs, the demanded minimum quality at each WPN might be possible to achieve with less SPs.

To define the objective function, an SPP is defined by a set of its WPNs W and a set of initial SPs $S^{\mathfrak{i}}$. Using both, along with the quality function (Equation 25), the maximum quality at every WPN can be calculated as

$$\begin{aligned} Q^{\max} = \{\max\{q(\mathbf{s}_{\mathfrak{h}}, \mathbf{s}_{\mathfrak{i}}, \boldsymbol{w}_{\mathfrak{j}}) \mid \\ (\mathbf{s}_{\mathfrak{h}} \in S^{\mathfrak{i}}) \wedge (\mathbf{s}_{\mathfrak{i}} \in S^{\mathfrak{i}}) \wedge (\mathbf{s}_{\mathfrak{h}} \neq \mathbf{s}_{\mathfrak{i}})\} \mid (\boldsymbol{w}_{\mathfrak{j}} \in W)\}. \end{aligned} \tag{77}$$

An essential part of the SPP is that the minimum quality is provided throughout the environment and therefore at each WPN. Therefore, for every WPN at which $Q^{\max}$ is less than the minimum quality, a penalty $\mathfrak{w}$ is applied. For the other WPNs the maximum quality at this point is used.

The resulting fitness function can be stated as

$$f^c = \sum \{q_i \mid (q_i \in Q^{max}) \wedge (q_i \geqslant \mathfrak{q})\} + \sum \{\mathfrak{w} \mid (q_i \in Q^{max}) \wedge (q < \mathfrak{q})\} \tag{78}$$

Finally, the constant $\mathfrak{w}$ has to be chosen in a way, which ensures that the fitness value for a solution that provides minimum quality at each WPN is always greater than a fitness value for a solution that does not.

Lemma 7. *For $\mathfrak{w}$ a value of $\mathfrak{w} < -|W|$ ensures that the fitness value of a solution that does not provide $\mathfrak{q}$ at every workspace point cannot be higher as a solution which does.*

Proof. The maximal fitness value for the proposed quality function is $|W|$, because the qualities are defined to be in range $[0, 1]$. Comparing the penalty free solution with the lowest fitness value to the penalized solution with the highest fitness value. The lowest penalty free solution can be obtained when all WPNs are covered with the minimum quality $\mathfrak{q}$. The highest penalized solution can be obtained when all WPNs but one are covered with the highest possible quality and one is covered with a quality just below the minimum quality $\mathfrak{q}$. This leads to the equation

$$\sum_{1}^{|W|} \mathfrak{q} > \left(\sum_{1}^{|W|-1} 1 \right) + \mathfrak{q} - |W|, \tag{79}$$

which can be simplified to

$$\sum_{1}^{|W-1|} \mathfrak{q} > \left(\sum_{1}^{|W|-1} 1 \right) - |W|.$$

Here, the term on the right side is negative because

$$\left(\sum_{1}^{|W|-1} 1 \right) = |W| - 1$$

and therefore

$$\sum_{1}^{|W-1|} \mathfrak{q} > |W| - 1 - |W| \Leftrightarrow -1.$$

Since the definition of the qualities implies that

$$\sum_{1}^{|W-1|} \mathfrak{q} \geqslant 0,$$

Equation 79 is true and $-|W|$ is a valid penalty for the fitness function. □

Because the quality function is based on the sine of the angle between two SPs that cover a WPN, the fitness function is nonlinear. The time complexity of calculating a fitness value is dominated by the calculation of all quality values in Q^{max}. For the calculation, the unique combinations of all SPs in S^i have to be build and then the quality has to be calculated for every unique combination with every WPN. Since the number of unique combinations can be calculated as $\frac{1}{2}(|S^i|^2 - |S^i|)$, the time complexity of evaluating the fitness function is

$$\mathcal{O}(|S^i|^2|W|). \tag{80}$$

6.3 MINIMUM QUALITY COVERAGE MODEL

In contrast to the maximum quality model, the minimum quality coverage model has the goal to ensure that the whole environment is covered and not just the defined WPNs. Instead of WPNs at which the quality is evaluated, the model uses the contour of the quality to define a sufficient quality region for each SC. The union of these regions is the overall percentage of the uncovered area, which serves as the fitness value during the optimization procedure.

Using Equation 22, a contour $c^q \in [0, 1]$ can be defined as

$$c^q = 1 - \frac{d^n_{i,j} \cdot d^n_{h,j}}{r \sin(\phi) q^{scale}}. \tag{81}$$

A visualization of different contour lines is shown in Figure 23 for two different sensor distances. As expected, a greater distance between both sensors leads to smaller contours because of the increase of at least one of the distances $d^n_{i,j}$ or $d^n_{h,j}$ at any point $\mathbf{p}_j$ between the two sensors $\mathbf{p}_i$ and $\mathbf{p}_h$ as shown in Figure 24.

The contour is a boundary that separates all points $\mathbf{p}^+$ with a quality $q^+ > c^q$, from points $\mathbf{p}^-$ with a quality $q^- < c^q$. Furthermore, the boundary either defines two or four coherent $\mathbf{p}^+$ regions, which can be seen in Figure 23 (two regions), and Figure 25 (four and two regions).

Because of their coherence, the $\mathbf{p}^+$ regions are qualified to be represented by one or four 2d polygons. Nevertheless, a polygon approximation of the contour is quite costly because it is based on numerous function evaluations to cover and refine the polygon edges. Therefore, constructing a fitness function on such a calculation will most likely make the optimization model insolvable in acceptable time. An alternative approach is to pre-calculate the contour polygons and use a lookup to determine the right contour polygon for an SC. A prerequisite for this approach is that the contour of an SC only depends on the relative position of the SPs, not on their absolute ones, which is shown in the following.

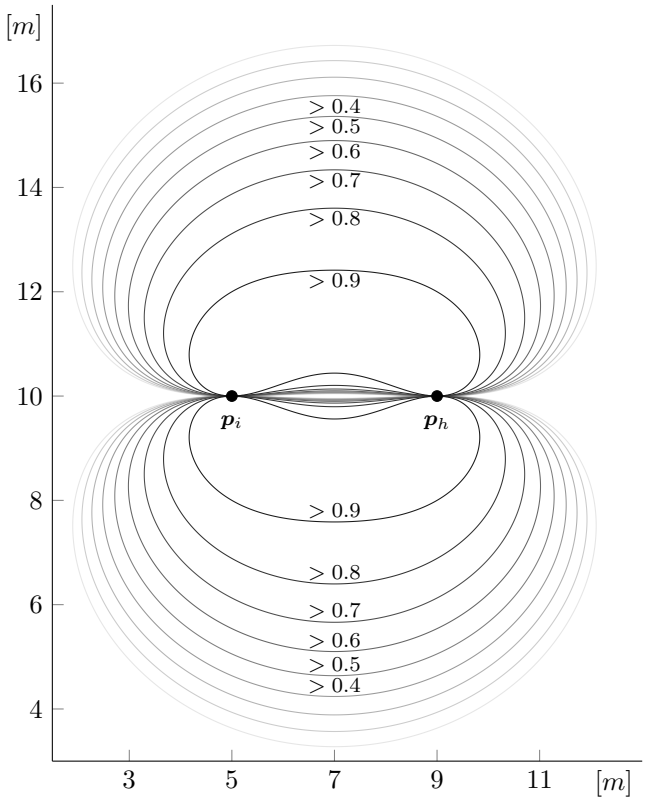


Figure 23: Contour lines of two sensors at positions $\mathbf{p}_i$ and $\mathbf{p}_h$ with a distance of 4 m. The lines are drawn at $q = \{0.1, \ldots, 0.9\}$, and their regions are labeled accordingly. As the labels indicate, the inner region of each contour contains values that are greater than the respective contour line.

To convert a contour to a polygon, the properties of the contour can exploited. In detail, the contour regions are symmetrical with a symmetry axis along the line $\mathbf{l}$ defined by the two points of the SCs h and i, shown in Figure 24, as

$$\mathbf{p}_i = (x_i, y_i) \tag{82}$$

$$\mathbf{p}_h = (x_h, y_h) \tag{83}$$

and another symmetry axis along the line that intersects

$$\mathbf{p}^{\text{mid}} = \mathbf{p}_i + \frac{(\mathbf{p}_h - \mathbf{p}_i)}{2}$$

in direction normal to $\mathbf{l}$. This symmetry can be derived by transforming Equation 20.

At first, the edges $(\mathbf{p}_i, \mathbf{p}_h)$, $(\mathbf{p}_h, \mathbf{p}_j)$ and $(\mathbf{p}_j, \mathbf{p}_i)$ between the two SPN $\mathbf{p}_h$, $\mathbf{p}_i$ and a WPN $\mathbf{p}_j$ always form a triangle in the 2d environment. The inner bearing angle $\sin(\gamma_{hij})$ can therefore be expressed using the law of sines

$$\sin(\gamma_{hij}) = \frac{\|(\mathbf{p}_i - \mathbf{p}_h)\|}{D}, \tag{84}$$

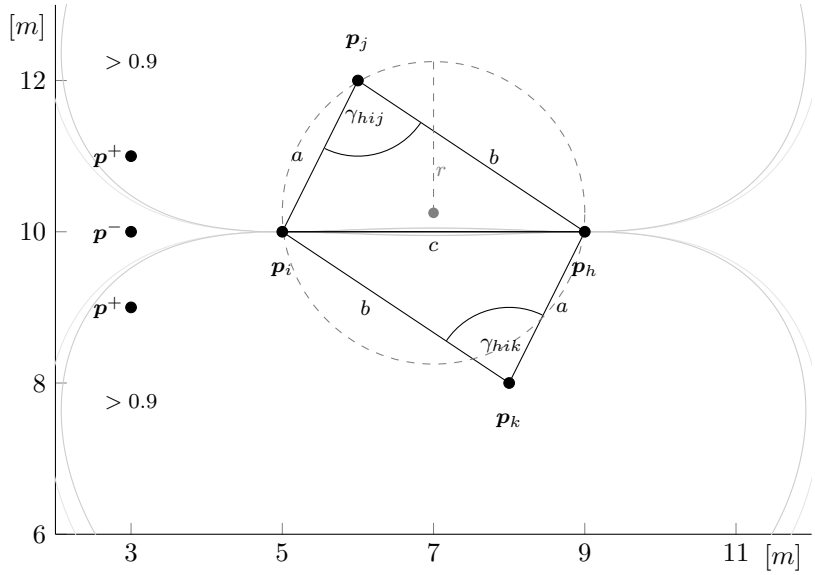


Figure 24: An excerpt of a continuous placement. The two points p_i and p_h represent the sensor locations and the points p_j and p_k represent two points in the environment. In addition, the lengths a, b and c are shown. The gray circle represents the circumcircle of the triangle p_i, p_h and p_k. The inner bearing angle γ_{hij} and γ_{hik} are shown along with three points, that are examples for points in a $q^+ > c^q$ and $q^- < c^q$ region.

whereas D is the diameter of the triangles circumcircle. Transferred to the quality calculation, the circumcircle can be calculated as

$$\begin{aligned} a &= \|(p_i - p_j)\| \\ b &= \|(p_h - p_j)\| \\ c &= \|(p_i - p_h)\| \\ D &= \frac{2abc}{\sqrt{(a+b+c)(-a+b+c)(a-b+c)(a+b-c)}}. \end{aligned} \tag{85}$$

A graphical representation can be found in Figure 24.

In conclusion, the inner bearing angle can be calculated using only the length of the triangle sides, which is related to the "Bogenschnitt", an arc intersection problem (Witte & Sparla, 2011). The same is true for the denominator of Equation 20, since it only depends on the length of $\|(p_h - p_j)\|$ and $\|(p_j - p_i)\|$. The used lengths a, b and c are shown in Figure 25. The Figure also shows the two symmetry axes at 9 m on the abscissa and 10 m on the ordinate. Both axes form a local coordinate system, in which every point

$$p^l := \begin{pmatrix} x^l \\ y^l \end{pmatrix} \tag{86}$$

is defined relative to the origin at

$$p_o := p_i + \frac{(p_h - p_i)}{2} = p_h + \frac{(p_i - p_h)}{2}. \tag{87}$$

The absolute coordinates of the origin are $x = 9\,m$ and $y = 10\,m$. The depicted points can defined in the global coordinate frame using their local coordinates

$$\mathbf{p}_1 \coloneqq \mathbf{p}_o + \begin{pmatrix} x^l \\ y^l \end{pmatrix}, \tag{88}$$

$$\mathbf{p}_2 \coloneqq \mathbf{p}_o + \begin{pmatrix} -x^l \\ y^l \end{pmatrix}, \tag{89}$$

$$\mathbf{p}_3 \coloneqq \mathbf{p}_o + \begin{pmatrix} -x^l \\ -y^l \end{pmatrix} \quad \text{and} \tag{90}$$

$$\mathbf{p}_4 \coloneqq \mathbf{p}_o + \begin{pmatrix} x^l \\ -y^l \end{pmatrix}. \tag{91}$$

To show their symmetry mathematically, the calculation of a and b is expanded for $\mathbf{p}_1$ using Equation 87 and 88 as

$$\begin{aligned} a &= \|\mathbf{p}_h - \mathbf{p}_1\|, \\ &= \left\| \mathbf{p}_h - \left(\mathbf{p}_h + \tfrac{(\mathbf{p}_i - \mathbf{p}_h)}{2} + \begin{pmatrix} x^l \\ y^l \end{pmatrix} \right) \right\|. \end{aligned} \tag{92}$$

With Equation 82, 83 and 88, Equation 92 becomes

$$a = \left\| \tfrac{1}{2} \left(\begin{pmatrix} x_i \\ y_i \end{pmatrix} - \begin{pmatrix} x_h \\ y_h \end{pmatrix} \right) - \begin{pmatrix} x^l \\ y^l \end{pmatrix} \right\|$$

and therefore,

$$a = \sqrt{\left(\frac{x_i - x_h}{2} - x^l \right)^2 + \left(\frac{y_i - y_h}{2} - y^l \right)^2}. \tag{93}$$

Analogue, using Equation 89, 87

$$\begin{aligned} b &= \|\mathbf{p}_i - \mathbf{p}_1\| \\ &= \|\mathbf{p}_i - \left(\mathbf{p}_i + \frac{(\mathbf{p}_h - \mathbf{p}_i)}{2} + \begin{pmatrix} x^l \\ y^l \end{pmatrix} \right)\| \\ &= \sqrt{\left(\frac{x_h - x_i}{2} - x^l \right)^2 + \left(\frac{y_h - y_i}{2} - y^l \right)^2}. \end{aligned} \tag{94}$$

Equation 93 and 94 can be simplified by using

$$d^x = \frac{x_i - x_h}{2},$$
$$-d^x = \frac{x_h - x_i}{2},$$
$$d^y = \frac{y_i - y_h}{2},$$

and

$$-d^y = \frac{y_h - y_i}{2}.$$

This leads to

$$a = \sqrt{(d^x - x^l)^2 + (d^y - y^l)^2}$$

and

$$b = \sqrt{(-d^x - x^l)^2 + (-d^y - y^l)^2}.$$

Conclusively, if either x^l or y^l changes its sign a will increase the same magnitude as b decreases.

By using Equation 85 in 84

$$\sin(\gamma_{hij}) = \frac{\sqrt{(a+b+c)(-a+b+c)(a-b+c)(a+b-c)}}{2ab}$$

and

$$q^c(a,b,c) = 1 - \frac{2a^2b^2}{\sqrt{(a+b+c)(-a+b+c)(a-b+c)(a+b-c)}},$$

the geometric quality in Equation 25 can be written as

$$q^r\left(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}_j\right) = \begin{cases} 0, & \text{if } d^n_{i,j} \vee d^n_{h,j} > 1, \\ q^c(\|\mathbf{s}_h, \boldsymbol{w}_j\|, \|\mathbf{s}_i, \boldsymbol{w}_j\|, \|\mathbf{s}_h, \mathbf{s}_i\|), & \text{otherwise.} \end{cases}$$

The equation shows, that in addition to the symmetry, the shape of the quality contour only depends on the distance c of the two sensors $\mathbf{s}_h$, $\mathbf{s}_i$, not their absolute position. If only the location of the two sensors changes, the shape of the contour will be isometrically transformed according to the new sensor coordinates. In conclusion, to perform an efficient optimization based on the geometric quality, the contour polygons can be pre-calculated for distances within the sensor range and stored in a lookup table.

For this purpose, let $\Gamma(\mathbf{s}_i, \mathbf{s}_h, q)$ be a function that returns the contour polygon for a configuration of two sensors and a defined quality. Such a function may be implemented to work on a pre-calculated database of contour polygons by first performing a lookup of a matching contour multi-polygon P based on the distance $\|\mathbf{s}_h, \mathbf{s}_i\|$ and afterwards transforming P isometric to the sensor position. In addition, let

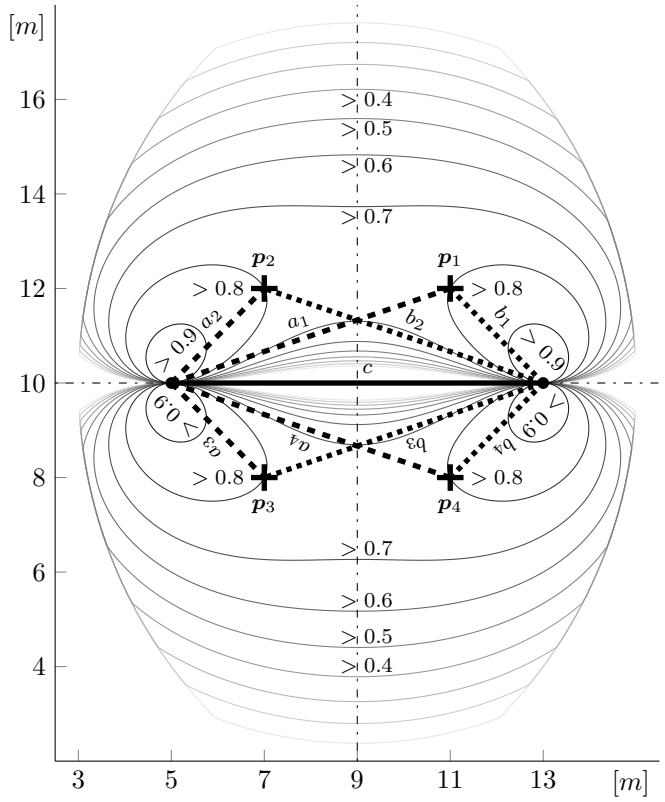


Figure 25: Contour lines of two sensors at a distance of 8 m. The lines are drawn at $q = \{0.4, \ldots, 0.9\}$, and their regions are labeled accordingly. As the label indicate, the inner region of each contour contains values that are greater than the respective contour line. From the outer regions of the contour lines, it can be seen that the contours < 0.6 are clipped by the detection range of the sensors. In addition to the contours, the Figure shows the symmetry by highlighting four points ($\mathbf{p}_1, \ldots, \mathbf{p}_4$) that have the same quality.

S^t be a configuration of SPs. By using $\Psi_{1,2}$ to include only the part of the contour that lies within the intersection of the VFOVs, the fitness function of the minimum quality coverage model can be stated as

$$f(S^t) = 1 - \frac{\|\mathfrak{P} \sqcap \left(\bigsqcup \left(\Gamma(\mathbf{s}_i, \mathbf{s}_h, q) \sqcap \Psi_{h,i}\right), \forall \mathbf{s}_i, \mathbf{s}_h \in S^t, \mathbf{s}_i \neq \mathbf{s}_h\right)\|}{\|\mathfrak{P}\|}. \quad (95)$$

In detail, this is the intersection of the environment with the union of all visible geometric quality contour polygons, which gives the covered region. The size of the area of this region is then divided by the size of the environment area to calculate the ratio of covered space. One minus this ratio gives the ratio of uncovered space that is to be minimized in the optimization process.

6.4 ITERATIVE CONTINUOUS POSITIONING SCHEME

The advantage of the continuous optimization methods is that they do not relay on pre-calculated sensor poses. Nevertheless, they are

only suited if the number of sensors is known in advance or at least an estimation of it exist. Here the discrete methods come into play since they provide a method to calculate an adequate number of sensors and their poses. Both methods can be combined to form an optimization strategy. For example, a discrete method or approximation is used to compute an initial placement solution. Then, this solution is taken as the initial guess to perform a continuous optimization.

Since the goal is to decrease the number of sensors, the maximum quality model can be exploited to try to find a valid solution when one of the sensors is removed. A valid solution is found if the fitness value drops below zero because no penalty is applied. If a valid solution cannot be found, the previous solution of optimized SPs is the improved quality at each WPN. In contrast, the minimum quality coverage model can be used in the same way if the goal is to maximize the covered area with the given number of sensors.

Both approaches are possibilities to apply the continuous models. Nevertheless, they require a non-linear optimization problem to be solved in each step, which in turn includes many graphical computations to calculate the fitness value. Hence, they are not likely to supersede the discrete optimization models in terms of solving time.

7

PLACEMENT EXPERIMENTS AND RESULTS

To conclude the work on SPPs, the conducted experiments and their results are presented. The experiments were conducted on digitalized floor plans of real world environments (Section 7.2). Their purpose is to test the proposed methods and compare their outcomes. The evaluation is centered on the solution quality in terms of the number of selected sensors compared for different solution strategies (models), environments and discretizations. It is divided into a meta-analysis that focuses on giving an overview of the findings (Section 7.5) and an exploratory data analysis that was performed on the obtained results to gain an improved understanding of the outcomes and a confidence in the applicability of the different algorithms (Section 7.6). Finally, a short summary of the experiments is given in Section 7.7.

The basis for all experiments, the configuration of the experimental setup, is introduced in Section 7.1. It introduces all necessary parameters and decisions to perform an SPP experiment.

7.1 PRACTICAL CONSIDERATIONS

Before the experiments are introduced, some parameter decisions that concern their realization are explained. In general, the parameters are based on practical considerations with respect to a real world setup of the ThILo system, for which the algorithms were developed.

Decision 1. *Fixed-point integer arithmetic with an accuracy of 1 mm is used for all geometric calculations.*

This decision is because floating-point accuracy is a major problem when dealing with geometric computations (de Berg, van Kreveld, & Overmars, 2008). Floating point values are not truly continuous but have a resolution that relates to the magnitude of the number. Hence, the underlying grid that represents the lowest possible resolution increases for a floating-point value with the magnitude of the number and so does every calculation result. In contrast, fixed-point integer arithmetic is based on a well-defined grid and provides stable geometric computations (Goodman & O'Rourke, 2004).

Another option to perform stable arithmetic computations is the usage of adaptive precision formats as exemplary stated by Shewchuk (Shewchuk, 1997). Here, the resolution is increased based to the performed calculations. Nevertheless, the disadvantage of this method is the increased computational complexity because there is no additional gain from calculating sensor placements and visibilities in sub millimeter range, these methods were not explored.

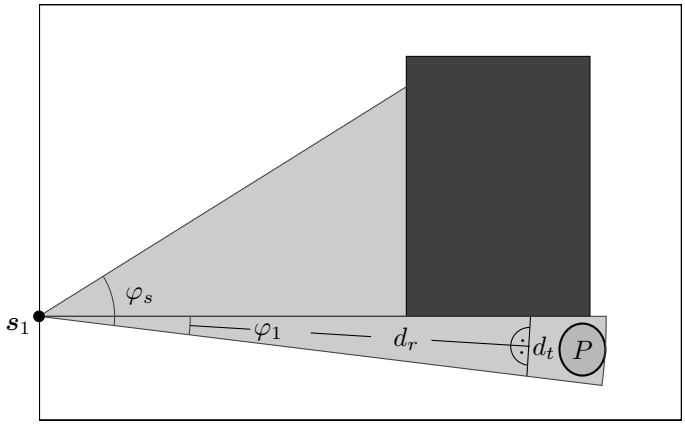


Figure 26: The VFOV of an SP s_1 is blocked partially by an obstacle and therefore contains a spike with an angle of φ_1. Within the spiked region, a person P is depicted as a circle. In addition, the length d_t of a tangent at distance d_r from the sensor is shown.

Decision 2. *All spikes in VFOVs that have an angular spike range of* $\mathfrak{s} \leqslant 3°$ *with respect to the SPN are removed.*

This decision is also based on the real world properties of the SPP. The measured value of each pixel is the received infrared radiation over its field of view. The system is used to locate the thermal radiation of humans, which have a torso diameter of approximately 0.5 m (Vereinigung der Metall-Berufsgenossenschaften, 2001). Therefore, the tangent of a spike as shown in Figure 26 should be large enough to allow a human to stand at the end of the detection range. If the human is approximated by a circle with a diameter of 0.5 m a tangent at 9.75 m would run through the center of the circle. To find minimal angle φ_1 for which the tangent is 0.5 m the equation

$$0.5\,\mathrm{m} = 2\sin(\varphi_1)9.75\,\mathrm{m},$$

has to be solved for φ_1. Here,

$$\sin\frac{\varphi_1}{2}9.75\,\mathrm{m} = \frac{d_t}{2}$$

as it can be observed from the geometric properties in Figure 26. This leads to

$$\varphi_1 = 2\arcsin\frac{0.5\,\mathrm{m}}{2\cdot 9.75\,\mathrm{m}} = 2.94°.$$

Therefore, an angular spike range of

$$\mathfrak{s} = 3° \tag{96}$$

is chosen as an overestimation, which results in a tangent length of

$$2\sin(1.5°)9.75\,\mathrm{m} = 0.52\,\mathrm{m}, \tag{97}$$

large enough to fit a human at a distance of 9.75 m.

Decision 3. *An initial grid size of* $\mathfrak{g} = 1\,\mathrm{m}$ *was used along with an angular resolution of* $\mathfrak{a} = 10°$ *for the discretization.*

The grid size and the angular resolution are based on the size of the evaluated environments. The grid size gives an initial distribution of WPNs that provides a basic quality measure in all rooms of every floorplan. The angular resolution is based on the sensor properties of the ThILo system, which provide a FOV of $> 45°$. Thus, on a right angle corner five SPs will be sampled.

The combination of both properties gives a coarse initial discretization in the small environments that can be increased up to ten times the initial value before the optimization of the MSPQM takes too long to finish in the given time box. Here, a substantial set of global optimal results could be computed for comparison with the other methods. In addition, the values are large enough that results for the MSPQM could be computed for the larger environments.

Decision 4. *A time box of 8 h was set to limit the runtime of the discrete optimization models.*

The decision is based on the number of SPPs that were solved for the experiments. Here, 8 h was a tradeoff that allowed a sufficient number of optimizations to be computed (> 10000) in a reasonable time of two month.

Decision 5. *A minimum quality of* $\mathfrak{q} = 0.45$*, which relates to a polygon area of approximately 0.6 m² is used.*

The decision is based on the real world properties of the ThILo system that was developed to serve as a positioning system for Ambient Assisted Living applications (Hauschildt & Kirchhof, 2010). It is especially suited to track the behavior of up to three humans in assisted living homes as shown in (Hauschildt & Kirchhof, 2011). Therefore, the positioning should enable behavior analysis that can recognize different actions based on the person's location and other context information. For instance, a person that is in front of the counter top at noon for longer than a few minutes is most likely to prepare lunch. Recently, deep learning algorithms have been proposed to solve such tasks (Längkvist, Karlsson, & Loutfi, 2014).

To provide location data that supports a recognition of activities, a positioning accuracy that supports discrimination of different activities that might occur in spatial neighborhood is needed. Examples are the different activities that take place very close to each other in the kitchen. Washing the dishes in the sink or cooking a meal on the stove right next to it might be less than a meter apart. To provide a sufficient quality for activity recognition tasks, the human torso was used as a reference for the minimum positioning quality. It approximately conforms to a circle with a diameter of 0.5 m (Vereinigung der Metall-Berufsgenossenschaften, 2001). If the arm length is considered

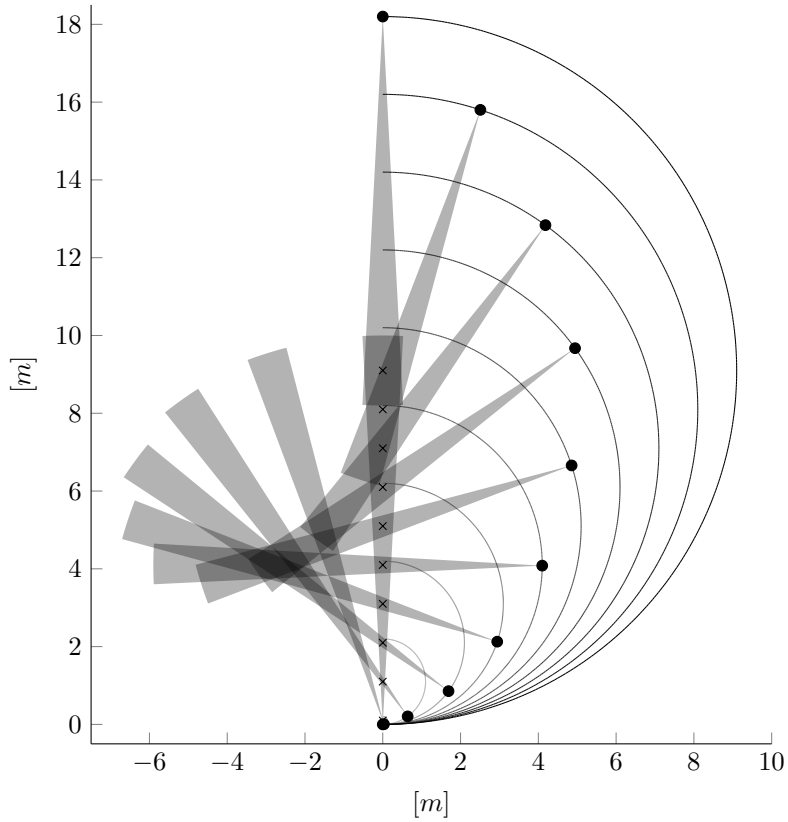


Figure 27: The setup for polygon intersection size and maximum interpoint distance evaluations. The circular lines on the right are the paths that allow a constant distance between the SP at $(0, 0)$, any SP on the path and the respective target in the middle of the VFOV marked with a "x". For the sensor at $(0, 0)$, the semitransparent gray cone represents the angular measurement error of $\pm 1.5°$, which is limited by the sensor range. For the different paths, some sensor poses are highlighted and their error cones are drawn. The overlay of the error cones also allows an easy deduction of the uncertainty region at this point.

the diameter increases to 0.8 m. Both measures served as references to choose an appropriate size of the uncertainty area, which was set to 0.6 m². This size conforms to a side length of 0.77 m if the area is assumed a rectangle and 0.87 m if it is assumed a circle. Thus, it slightly underestimates the distance for rectangular shaped error polygons and slightly overestimates it for circular shaped ones.

To define a GDOP threshold based on this requirement, the sensor's angular detection error and maximum detection range has to be taken into account. For the ThILo system, which uses sensors with a FOV of 48°, the angular error was determined in (Kemper & Hauschildt, 2010) to be $\pm 1.5°$, whereas the detection range was determined to be approximately 10 m. With these parameters, the area and the interpoint distance of the intersecting uncertainty polygon can be

calculated for each WPN. To compare these to the GDOP values, an additional scaling factor was introduced for Equation 20 as

$$u^{sc}(\mathbf{s}_h, \mathbf{s}_i, \boldsymbol{w}) = u^{scale} \frac{d^n_{i,j} \cdot d^n_{i',j}}{\sin(\phi)}. \tag{98}$$

In Figure 28 the area and in Figure 29 the maximum interpoint distance of the uncertainty polygon are plotted against the scaled GDOP quality values for increasing distances over a range of $[0°, 180°]$ for the inner bearing angle. The setup that was used is shown in Figure 27. It allows the calculation of uncertainty intersection polygons and their maximum interpoint distance for sensor configurations where both sensors are equally distant to the target.

To compare the quality values to the polygon areas, the u^{scale} value was calculated that minimizes the error between both functions in the most relevant range of $[20°, 160°]$. This led to a scale $u^{scale} = 1.1$ for the polygon size and $u^{scale} = 2.44$ for the maximum interpoint distances. The shape of the GDOP represents the polygon size quite well opposed to the maximum interpoint distance. For latter, only the general behavior is met. The GDOP exceeds the maximum interpoint distances if the object is far from the sensors and falls behind on it if the object is closer to the sensors. Nevertheless, the GDOP generally overestimates the maximum interpoint distance and therefore placement solutions based on it are likely to have a better maximum interpoint distance.

With the defined requirement of a maximum uncertainty area of $0.6\,m^2$ and by using the scaling factor of the fitted GDOP, the uncertainty threshold can be stated as

$$\begin{aligned} u &= \frac{0.6}{1.1} \\ &\approx 0.55. \end{aligned} \tag{99}$$

Using this threshold, the quality function 21 has a scale of

$$q^{scale} = 1. \tag{100}$$

In addition, the minimum quality of SP-SP-WPN configurations is defined as

$$\begin{aligned} q &= 1 - u \\ &\approx 0.45 \end{aligned} \tag{101}$$

7.2 INPUT ENVIRONMENTS

Beside the optimization and preprocessing parameters, the essential prerequisite for the SPP is a description of the environment where the positioning takes place. This description has to be provided in form of a 2d polygon, as described in Section 2.6.

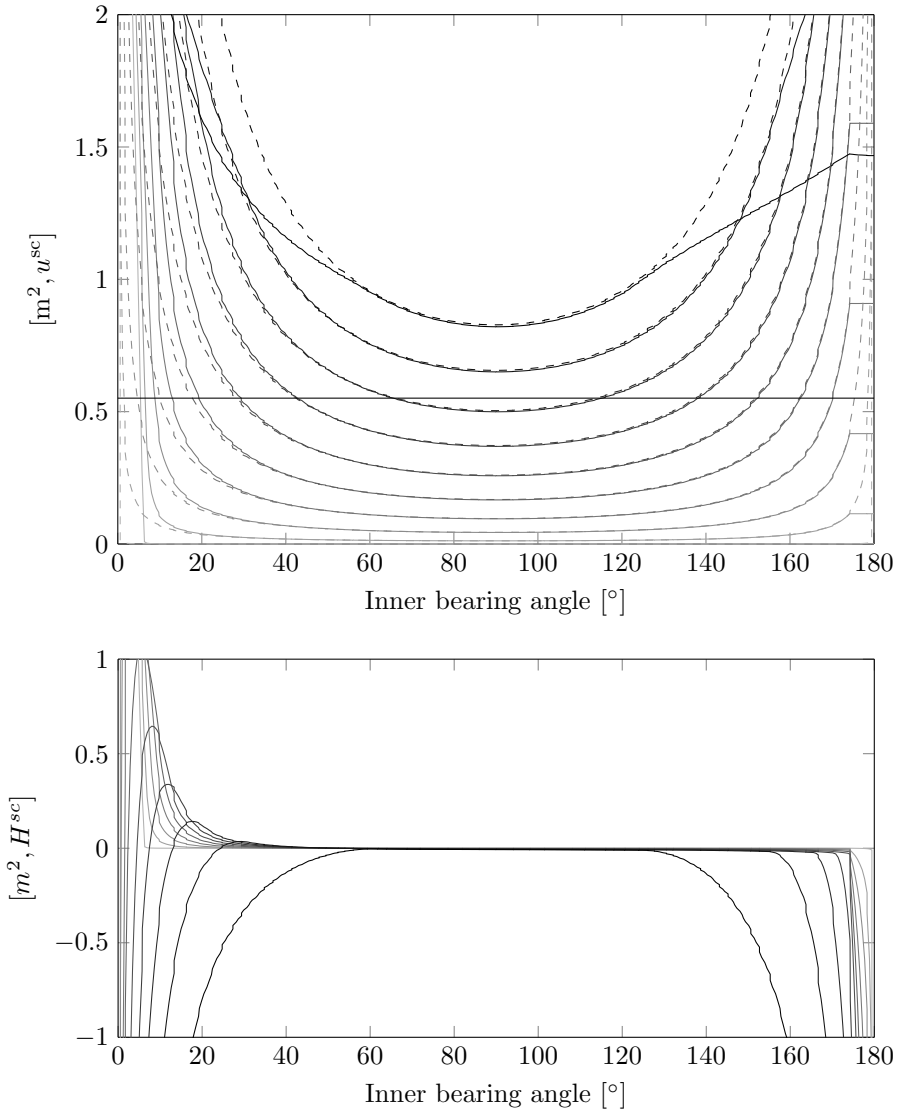


Figure 28: The top Figure shows the sizes of the uncertainty polygon intersections over the inner bearing angle for different distances in solid lines. The distances range from 0.2 m (light gray) in steps of 2 m to a maximum distance of 18.2 m (dark gray). The uncertainty polygons are based on an angular error of 3°. The dashed lines are the quality values, calculated using Equation 98 with $\mathfrak{r} = 10\,\mathrm{m}$ and $\mathfrak{u}^{\mathrm{scale}} = 1.1$. It can be observed that the quality values represent the sizes well, except close to the inner bearing angle limits (0°, 180°). The different behavior of the sizes calculated when both sensors are 18.2 m apart, is explained by the partial overlapping of their VFOV due to their distance.
The bottom Figure shows the differences of the sizes and qualities calculated individually for each distance. Each difference is calculated by subtracting the quality from the size for the respective distance. Thus, a positive difference shows an underestimation of the uncertainty polygon size, whereas a negative difference shows an overestimation.

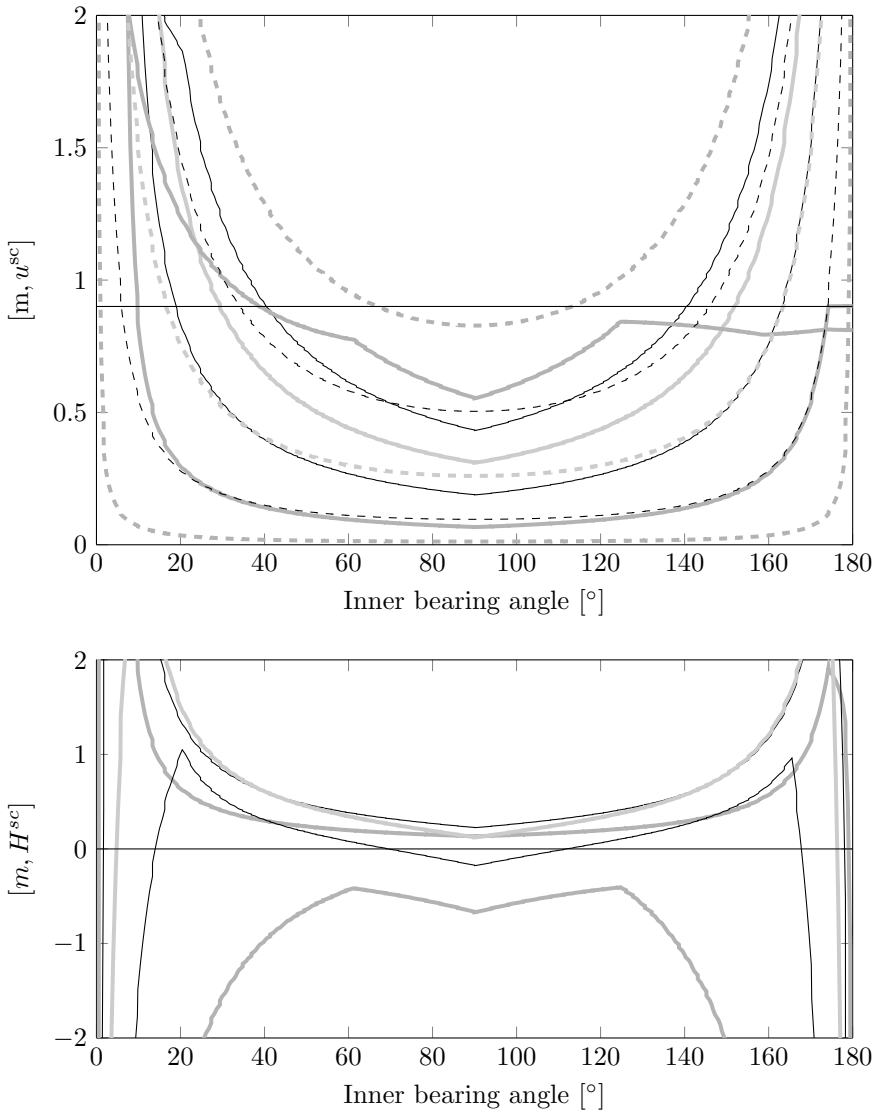


Figure 29: The top Figure shows the maximum interpoint distance over the inner bearing angle for different distances in solid lines. The distances range from 6.2 m in steps of 4 m to a maximum distance of 14.2 m. They are plotted in alternating black and gray lines. The uncertainty polygons are based on an angular error of 3°. The dashed lines are the quality values, calculated using Equation 98 with $\mathfrak{r} = 10\,\mathrm{m}$ and $\mathfrak{u}^{\mathrm{scale}} = 2.44$. In contrast to the polygon sizes, the max. interpoint distances are only matched well by the quality metric if the distance is not too far away from the sensors and the inner bearing angle is between 20° and 160°.
This can be well observed in the bottom figure, which shows the differences of the max. interpoint distances and qualities calculated for each distance. The difference is calculated by subtracting the quality from the size. Thus, a positive difference shows an underestimation of the uncertainty polygon size, whereas a negative difference shows an overestimation.

To show the working and scalability of the proposed algorithms, four different environments were evaluated. They represent typical indoor environments of different sizes. Their graphical representation in form of floor plans are shown in figures 30, 31, 32 and 33. The smallest is a single conference room with a size of 31 m² that includes a big table, a cupboard and a wall mounted white board. Further, a small apartment with a size of 35 m², a large apartment with a size of 81 m² and an office floor with a size of 354 m² are used for the evaluation. The office floor is by far the biggest environment and contains four offices, a conference room, a shared workspace and a computer lab.

The furniture that is placed in the environments can be deduced from the floor plans. The different classes of objects that were defined in the environment description can be distinguished by their inner and boundary color. Gray areas with gray borders represent occupied regions, dark gray areas with black borders are obstacles and mountables are represented by shapes with a gray filling and a black border. The assignment of the real world objects to the defined object classes was based on the properties of the ThILo. It operates on heat detecting thermopile sensors that are placed in a height of approximately 1.2 m. Thus, objects like tables, sideboards, chairs do not obstruct their FOV and are regarded as occupied regions. Higher objects are either obstacles or mountables based on their ability to be equipped with sensors.

The number of SPs and WPNs are the parameters of the experimental setup that were varied among the subsequent evaluations. To give an idea of the geometrical distribution of both parameters two discretization granularities are visualized in figures 30, 31, 32 and 33. The first one is the initial set of parameters with a WPN distance of 1 m and SPN at the convex environment corners. The second one shows a discretization with 200 additional workspace positions (AWPN) and 200 additional sensor poses (ASP). This was the largest amount of AWPN and ASP for which global optimal solutions could be computed within the predefined time box.

The initial number of number of workspace positions (#WPN) is 23 for the CR, 19 for the SF, 59 for the LF and 202 for the OF. This number is increased by up to 500 AWPN for each environment whereas every AWPN increases the #WPN, which in turn decreases the distance among the sampled points. The WPN sampling algorithm (Algorithm 1) initially samples a grid of 1 m, which is also the mean of minimum distances in case no WPN is isolated from its grid neighbors. In the CR, SF and LF this is the case. In contrast, the OF has some WPNs for which no direct neighbor on the initial grid exist and therefore the mean of minimum distances for this environment is slightly higher (1.012 m). One of these WPNs is the bottommost left one in Figure 32.

AWPN	CR	SF	LF	OF
0	1000 mm	1000 mm	1000 mm	1012 mm
100	317 mm	364 mm	550 mm	753 mm
200	263 mm	259 mm	404 mm	622 mm
500	152 mm	162 mm	260 mm	520 mm

Table 8: The mean of minimum distances for different environments and numbers of AWPN.

(ASP, AWPN)	CR	SF	LF	OF
(0, 0)	2849	3632	12333	216025
(200, 200)	677793	152771	169713	578240
(500, 500)	8164523	1414676	885475	1591610

Table 9: The number of SP-SP-WPN combinations for different environments and their number of ASP and AWPN.

A summary of the mean of minimum distances given in millimeter is shown in table 8. Here, it can be seen that the CR and SF have a similar overall behavior even though the distribution at 100 AWPN is not as uniform for the SF. The size of the LF and the OF leads to significantly bigger distances for both environments.

In contrast to the WPNs, the initial number of sensor poses (#SP) is not only depended on the environment properties but also on the distribution of WPNs because sampled SPs are removed by a preparation procedure if they do not cover any WPN. Thus, with an increasing number of WPNs the initial #SP might seem to increase because less SPs are removed. With no AWPN, the number of initially sampled SPs was 38 for the CR, 135 for the SF, 251 for the LF and 991 for the OF. The CR is the only environment where the #SP is constant even if the #WPN increases. For the SF, 16 SPs are filtered if the #WPN is below 119. The #SP for the LF drops to 249 if the #WPN is below 149 and 238 if the #WPN is below 49. The #SP of the OF is 941 for $\leqslant$ 202 WPNs, 959 for $\leqslant$ 302 WPNs and 984 for $\leqslant$ 402 WPNs.

Beside the independent consideration of the SPs and WPNs another important factor is the number of SP-SP-WPN combinations, which gives a hint to the computational complexity of finding the optimal placement solution.

In table 9, it can be seen that the increase is opposing to the size of the environment and especially the obstacles and mountables, which block the sensors visibilities. In the CR each new SP leads to a new SP-SP-WPN combination with most of the already sampled SPs and most of the WPNs. In the SF and LF the creation of new combinations is restricted to the rooms because their walls limited the visibilities.

This is also true for the OF but here the large computer pool raises the number of combinations significantly.

7.3 DECOMPOSITION

Each environment was decomposed into convex polygons using the RCPD algorithm. Along with the introduced edges that define the convex pieces, the WPNs and SPs were partitioned into sub-problems. The number of convex pieces that result in sub-problems depends on the distribution of WPNs because a sub-problem only exists if at least one WPN needs to be covered.

The overall number of convex pieces is 11 for the SF, 36 for the LF and 82 for the OF. The number of sub-problems, which depends on the number of sampled WPNs, is between 5 and 7 for the SF, 20 and 24 for the LF and 41 to 55 for the OF. The distribution of WPNs on the convex parts is quite similar for the SF and LF. Around 66% of the WPNs are contained in the largest 30% of the convex parts and 95% of WPNs are contained in the largest 70% of the convex parts. The distribution of WPNs in the OF is more towards the larger convex parts. Here, 66% of the WPNs are contained in the largest 20% of convex parts, whereas 95% of the WPNs are also contained in the largest 70% of convex parts.

Beside the objective evaluation, a visualization of the decompositions can be found in figures 30, 31, 32. Here it can be seen that the decomposition performs well in separating the different rooms from each other in all three environments. The separations inside the rooms that are due to the placed obstacles are sometimes disadvantageous. Especially in the LF, the rooms are usually separated into three or more parts that seem to belong together if analyzed visually. On the other hand, the convex parts in the computer lab (right part) of the OF serve as a positive example how to separate points in meaningful convex parts. Especially in the lower half, the straight edges provide equally sized regions.

However, these statements are only expert guesses and variations in convex decomposition are not evaluated by solving additional SPP because the focus of the evaluations is on the comparison of the different introduced methods, rather than the successive improvement of one of them.

7.4 MODELS AND EXPERIMENTAL SETUP

The models that are tested and compared with each other using the input environments are:

BSPQM	Best Sensor Pairwise Quality Model

CMQM	Continuous Maximum Quality Model
GCS	Greedy Combined Selection
GSSS	Greedy Single Sensor Selection
MSPQM	Minimum Sensor Pairwise Quality Model
RCPD-B	RCPD combined with BSPQM
RCPD-M	RCPD combined with MSPQM
SCO	Sensor Combination Optimization
STCM	Simple Two-Coverage Model

The greedy selection strategies are not models themselves but algorithms that compute valid solutions, nevertheless to simplify the naming they will be referred to as "models".

The two models RCPD combined with MSPQM (RCPD-M) and RCPD combined with BSPQM (RCPD-B) are the combination of the RCPD with the respective model applied to solve each of the subproblems defined by the convex parts. The solution was then assembled by an union of all selected sensors for every part. All models where evaluated for every environment under varying discretization sizes. The additional sensor poses (ASPs) and additional workspace positions (AWPNs) were varied from 0 to 500 with a step size of 10 for the greedy models and the STCM. Thus, 2601 SPPs were solved per model and environment, which makes a total of 41616 evaluations, not including the 10404 sensor coverage models that were solved as a perquisite for the GSSS placement.

The global optimization models MSPQM and BSPQM as well as the local optimization models RCPD-M and RCPD-B could not be evaluated this extensively because of their runtime. For both models, the 'CPLEX Optimization Studio 12.5'(IBM, 2015) was used. The upper limit on discretization accuracy was bounded by the models solving time. Therefore, the runtime of each optimizations was limited to 8 h using 10 'Intel Xenon E5649 cores' with 2.53GHz each and a total of 20GB available RAM. The solution process was started with 0 ASP and AWPN. Both values were increased with a step size of 10 for the MSPQM and RCPD-M as well as 50 for the BSPQM and RCPD-B to a maximum number of 200 ASP and AWPN. Due to the time constraints only the CR, SF could be fully evaluated with 441 solved MSPQM and RCPD-M as well as 25 solved BSPQM and RCPD-B evaluations. The same is true for the MSPQM, RCPD-M and RCPD-B in the LF. It was tried to solve the BSPQM for the LF with no ASP and AWPN without any time constraint but this approach was canceled after the optimization ran for a week and still had a gap of 5% left.

Thus, the BSPQM was not evaluated for the OF and for this environment there were also very few MSPQM, RCPD-M and RCPD-B solutions that could be computed in time. Even though, the evaluation of discrete models took approximately two months in which 1996 optimization models were solved.

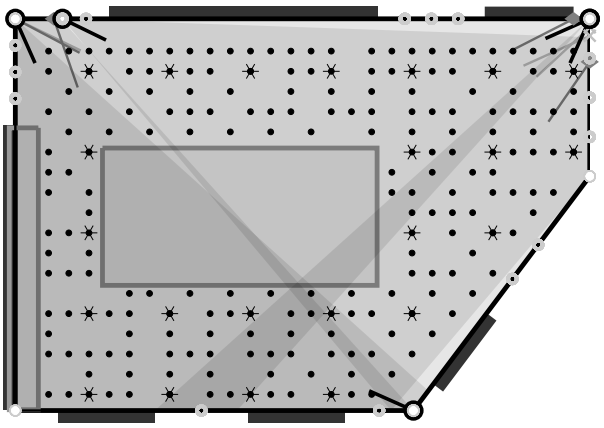


Figure 30: The initial placement of the 23 WPNs (spiked dots) and the placement with 200 AWPN is shown for the CR. The black SPs on the boundary, drawn as small circles with an outgoing line that indicates the center of the VFOV bearing, show the best MSPQM solution that uses five SPs to cover a discretization with 200 AWPN. The VFOV of the solution are drawn in a semitransparent gray to give an indication of the covered area. In addition, it shows the overlapping of multiple VFOV by the shade of gray. An example can be observed in the middle of the room, where the darker gray spot indicates an intersection of five VFOV, two from the sensors in the upper right corner, two from the sensors in the upper left part of the boundary and one from the sensor in the lower right corner. The four gray SPs on the boundary show the best continuous CMQM placement. Its computation was started with an initial set of eight SPs that were subsequently reduced to four resulting in a mean of max. quality of 79%. In addition, the light gray SPs mark the best CMQCM solution that covers the whole environment with a minimal quality using four SPs. Furthermore, the initial 5 SPN of the 38 SPs are drawn as white circles on the boundary along with the gray circles that represent the 15 SPN at 200 ASP.

In contrast to the other models that are extensively explored in the following sections, STCM will only be mentioned briefly. As opposed to the other models, the STCM does not include any quality metric other than the sameplace constraints. Therefore, the solution can be seen as a lower bound on how many SPs are at least needed to cover each WPN twice. Nevertheless, the interested reader finds some results of this model along with the extended results of the other models in the appendix Chapter A.

In contrast to the discrete models, the continuous models were solved by using two different methods. The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) proposed by Hansen and Ostermeier (Hansen & Ostermeier, 2001) along with a gradient decent interior point (IP) algorithm (Yao et al., 2008). Because both models are highly nonlinear, the goal of the evaluation was to test if they can be solved to a sufficient solution using off-the-shelf solvers. Both continuous models used the results of the discrete SCO optimization as initial state. The CMQM was then evaluated for an increasing number

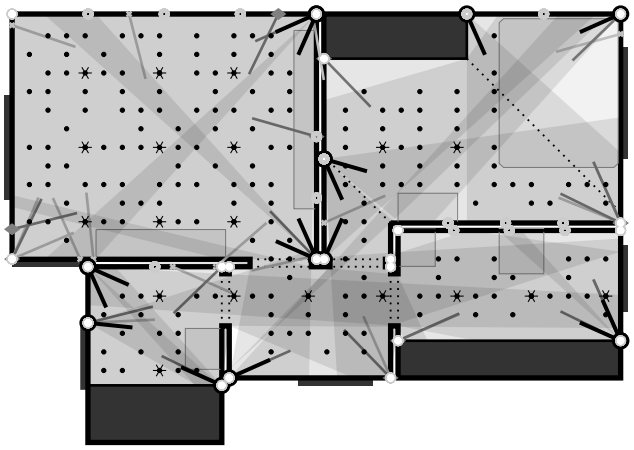


Figure 31: The initial placement of 19 WPNs and the placement with 200 AWPN for the SF. The black SPs on the boundary, show the best MSPQM solution that uses 16 SPs to cover a discretization with 200 AWPN. The 18 SPs in gray show the best continuous CMQM placement. Its computation was started with an initial set of 19 SPs which were subsequently reduced to four resulting in a mean of max. quality of 95%. In addition, the light gray SPs mark the best CMQCM solution that allows a coverage of 94% of the environment with a minimal quality using 18 SPs. Furthermore, the initial 16 SPNs of the 118 SPs are drawn as white circles on the boundary along with the gray circles that represent the 36 SPNs at 200 ASP. Finally, the 11 convex polygons calculated by the RCPD are shown using the black dotted lines. A detailed description of the visualized properties is given in Figure 30.

of AWPN with a step size of 50 up to 500 AWPN. In each optimization run, it was tried to find a valid solution that covers all WPNs with the desired minimum quality, which is a solution with an objective function value $\leqslant 0$ (see Section 4.2). If such a solution was reached, the number of SPs was reduced by one and the optimization was restarted. The time limit to find a valid solution was set to 4 h for each optimization run. Thus, whenever a solution was found the clock was reset. If no solution could be found, the last valid solution was returned.

The CMQCM optimization was only executed once for each environment using both solvers. This model does not depend on the WPNs, thus there was no need to try different WPN discretizations. The goal of the solving process was to find a coverage of at least 99% of the environment space with the desired quality. If such a solution was found, the optimization was restarted with one SP removed. As for the CMQM, the maximum time per optimization step was 4 h.

7.5 META ANALYSIS OF EXPERIMENTAL RESULTS

The empirical results are presented in a top down fashion. At first, the meta results will be presented as a statistical summary of the individ-

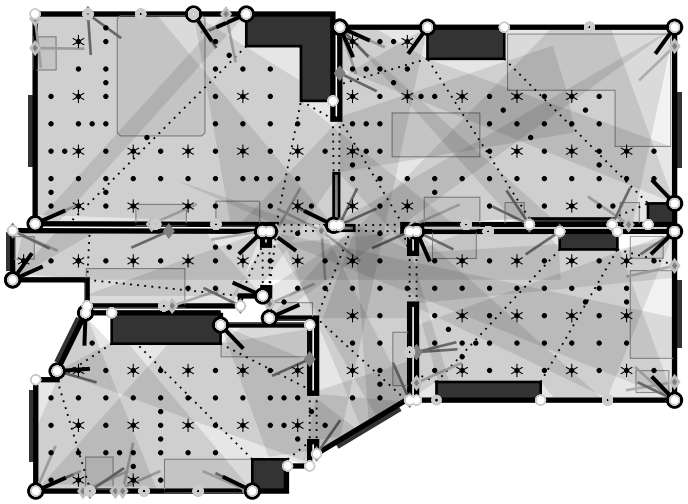


Figure 32: The initial placement of the 59 WPNs (spiked dots) and the placement with 200 AWPN is shown for the LF. The black SPs on the boundary show the best MSPQM solution that uses 23 SPs to cover a discretization with 200 AWPN . The 29 SPs in gray show a continuous CMQM placement. Its computation was started with an initial set of 34 SPs that were subsequently reduced to 29 resulting in a mean of max. quality of 88%. In addition, the light gray SPs mark the best CMQCM solution that allows a coverage of 98% of the environment with a minimal quality using 29 SPs. Furthermore, the initial 42 SPNs of the 238 SPs are drawn as white circles on the boundary along with the gray circles that represent the 57 SPNs at 200 ASP. Finally, the 36 convex polygons calculated by the RCPD are shown using the black dotted lines. A detailed description of the visualized properties is given in Figure 30.

ual optimization results, and then the evaluated environments will be compared in detail using an excerpt of the gathered data. Overall, the goal was to find relation between the size of a problem and its solvability as well as their performance relative to each other. Furthermore, the influence of the geometric properties of the environment was to be explored.

The software that was used to perform the experiments as well as all experimental data is available on the memory card that is enclosed with the dissertation. In addition, an up to date version of the software is available online at https://github.com/nicolajkirchhof/SensorPositioning.

The summary of results is presented in four sets of three plots each. One set contains the resulting number of SPs, the quality of the model measured in percentage and the percentage of covered area. The mean of the maximum quality at each WPN is used as the “quality” measure. This metric has the advantage that it does not directly depend on the number of selected SPs, because only the best quality instead of e.g. a sum of all qualities is used. Nevertheless, with

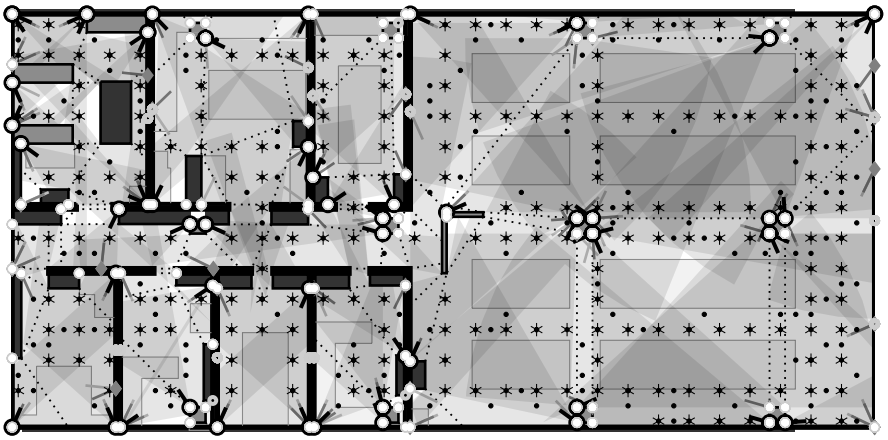


Figure 33: The initial placement of the 202 WPNs (spiked dots) and the placement with 200 AWPN is shown for the LF. The black SPs poses on the boundary show the best MSPQM solution that uses 59 SPs to cover a discretization with 200 AWPN. The 80 SPs in gray show a continuous CMQM placement. Its computation was started with an initial set of 82 SPs that were subsequently reduced to four resulting in a mean of max. quality of 83%. In addition, the light gray SPs mark the best CMQCM solution that allows a coverage of 92% of the environment with a minimal quality using 73 SPs. Furthermore, the initial SPNs of the 941 SPs are drawn as white circles on the boundary along with the gray circles that represent the SPNs at 200 ASPs. Finally, the 82 convex polygons calculated by the RCPD are shown using the black dotted lines. A detailed description of the visualized properties is given in Figure 30.

a greater #SP the quality at each point naturally increases due to the higher number of SCs.

The figures that show the statistics for the CR, SF, LF and OF are shown in 34, 35, 36, 37. Every Figure is a boxplot that shows the mean as a thick black bar, the range of the box begins at the 25^{th} percentile and stretches to the 75^{th} percentile. The whiskers show the extreme data points, while outliers are separately shown by a gray plus sign.

Every abscissa is labeled from A to I whereas each label correspond to one of the models as:

A	CMQM	(IP)	continuous models
B	CMQM	(CMA-ES)	
C	SCO	greedy	approximate models
D	GCS		
E	GSSS		
F	MSPQM		global models
G	BSPQM		
H	RCPD-M		decomposition
I	RCPD-B		

As expected, it can be seen that the approximate models (C, D and E) have a higher #SSP and provide a better quality than the global optimization models. The overall performance of the RCPD based optimization models is in range of the approximate models for the SF. For the LF and OF, the #SSP of the RCPD based approximation models rapidly increases and they provide the worst maximum and mean results of all models.

The poor performance of selected SPs of the RCPD-M for the LF and OF is due to the large amount of convex pieces that were created for these environments. Since the selected SPs are only optimal for the convex pieces, which are not well separated for the inner of the rooms, the result suffers from the lack of SPs that cover WPNs in multiple convex pieces.

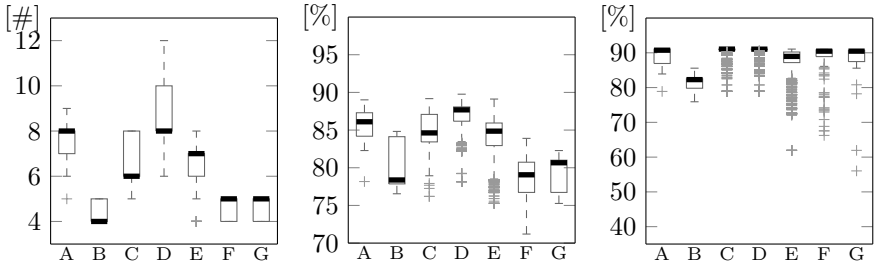
The CMQM models perform slightly better than the SCO, which indicates that they find at least some configurations where the initial number of sensors can be decreased. An exception is the CR. Here it can be seen that their results provide a significant decrease in #SSP, if the CMA-ES solver is used.

The percentage of covered area for the LF and OF is quite stable, whereas the CR and especially the SF show heavy fluctuations. Latter seem to be a result of the sparse environment discretization with only 19 initial WPNs. Especially the upper right room has only two WPNs in the middle that need to be covered, as shown in Figure 31.

In comparison with each other, the MSPQM and BSPQM (F, G) are quite similar in #SSP. In addition, the difference in quality is only marginal. For the LF, the MSPQM seems to outperform the BSPQM in terms of quality with a slightly worse #SSP. The three greedy models seem to follow a strict order in which the GSSS performs best, followed by the SCO and the GCS, which has the worst overall results in #SSP.

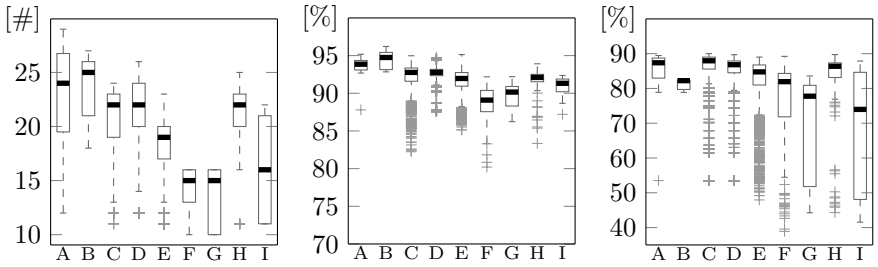
It has to be noted that the evaluation of the RCPD-B for the LF and OF contained a systematical error that lead to a selection of all SPs by the MIP solver for parts with only one or two WPNs. Because the RCPD-B for the SF along with the RCPD-M for the LF and OF contained enough information to provide thorough results, the erroneous RCPD-B results were excluded and not redone.

A comparison of the RCPD-M and RCPD-B statistics for the SF shows an expected result where the RCPD-B outperforms the RCPD-M in terms of selected sensors. However, this statistic is flawed because the RCPD-M was evaluated for a wider range than the RCPD-B. Since the range of the RCPD-B was the same as for the BSPQM and MSPQM these three are well comparable and show that the means of #SSP for all three models are with 14.1 (MSPQM), 13.6 (BSPQM) and 16.7 (RCPD-B) are quite close in contrast to their variance.



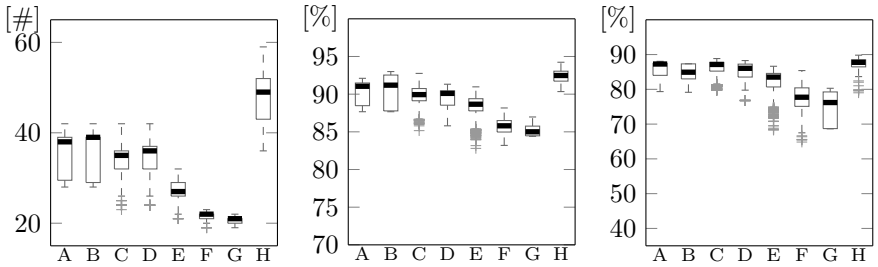
(a) #SSP (b) Quality (c) Area

Figure 34: Conference Room



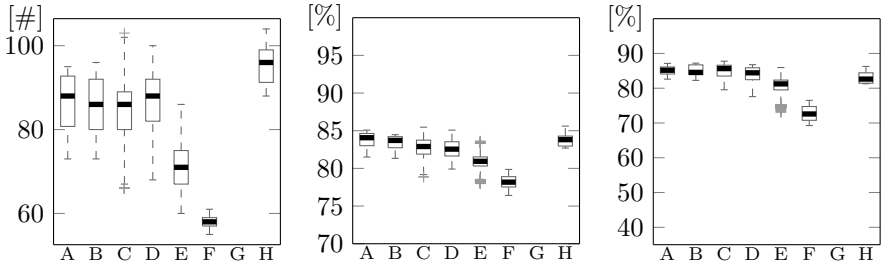
(a) #SSP (b) Quality (c) Area

Figure 35: Small Flat



(a) #SSP (b) Quality (c) Area

Figure 36: Large Flat



(a) #SSP (b) Quality (c) Area

Figure 37: Office Floor

7.6 DETAILED COMPARISON

The overall statistics give a good first impression of obvious correlations. Nevertheless, to represent the parameters of the experimental results in a more detailed way, multi bar plots were created for each environment. They show an excerpt of the CR, SF, LF and OF experiments in figures 38, 39 and 40[1]. Whereas the multi bar plots for the CR and SF contain results for almost all algorithms, the range of the multi bar plot for the LF was reduced to fit the MSPQM results that could be computed.

The multi bar plots provide insight in the most important parameters of the performed evaluations. Their outer axis represents the AWPN (#WPN) on the abscissa and the ASP (#SP) on the ordinate. Each bar plot within this coordinate system contains solution results of different models (as shown in the legend), whereas each solution result consists of four parameters: #SSP, quality, covered area and sum of qualities. All of the presented variables are scaled based on the global minimum and maximum of the respective parameter over all visualized experiments of this environment.

The most important parameter, the #SSP is represented by the gray box in the background. It is also written on the abscissa below the respective box to quantify the box. Within the lower half of the ordinate range (shown by dotted line), the percentage of covered area is visualized by a black bar that is labeled with the percentage value. The quality of the solution is presented in the upper half between the dotted line and the top of the ordinate. Each indicator is also labeled with its quality value. At last, the small dots on the bars are a representation of the sum of all qualities for every WPN. They provide a qualitative comparison possibility between two models of the same environment. All percentage values are rounded for presentation purposes.

7.6.1 *Selected Sensors and Quality*

The results of the optimization models show that the increase in #SSP is mainly related to the #WPN. The #SP, which gives the algorithms a wider range to place sensors, has almost no influence. Only the MSPQM and BSPQM solutions with 100 ASP and 0 AWPN for the LF use one less SP each. Nevertheless, the different percentages of covered area indicate that the #SP changes the outcome of the optimizations and leads to different SP selections. In detail, the #SSP for the CR only increases once for the global optimization models, when

1 In addition, Figure 45 and 46 in the appendix show the multi bar plot of approximate evaluations and a bar plot for the first of the OF evaluations. Both have been excluded from the thesis main part because they do not provided additional information other than the overall value range of the solutions.

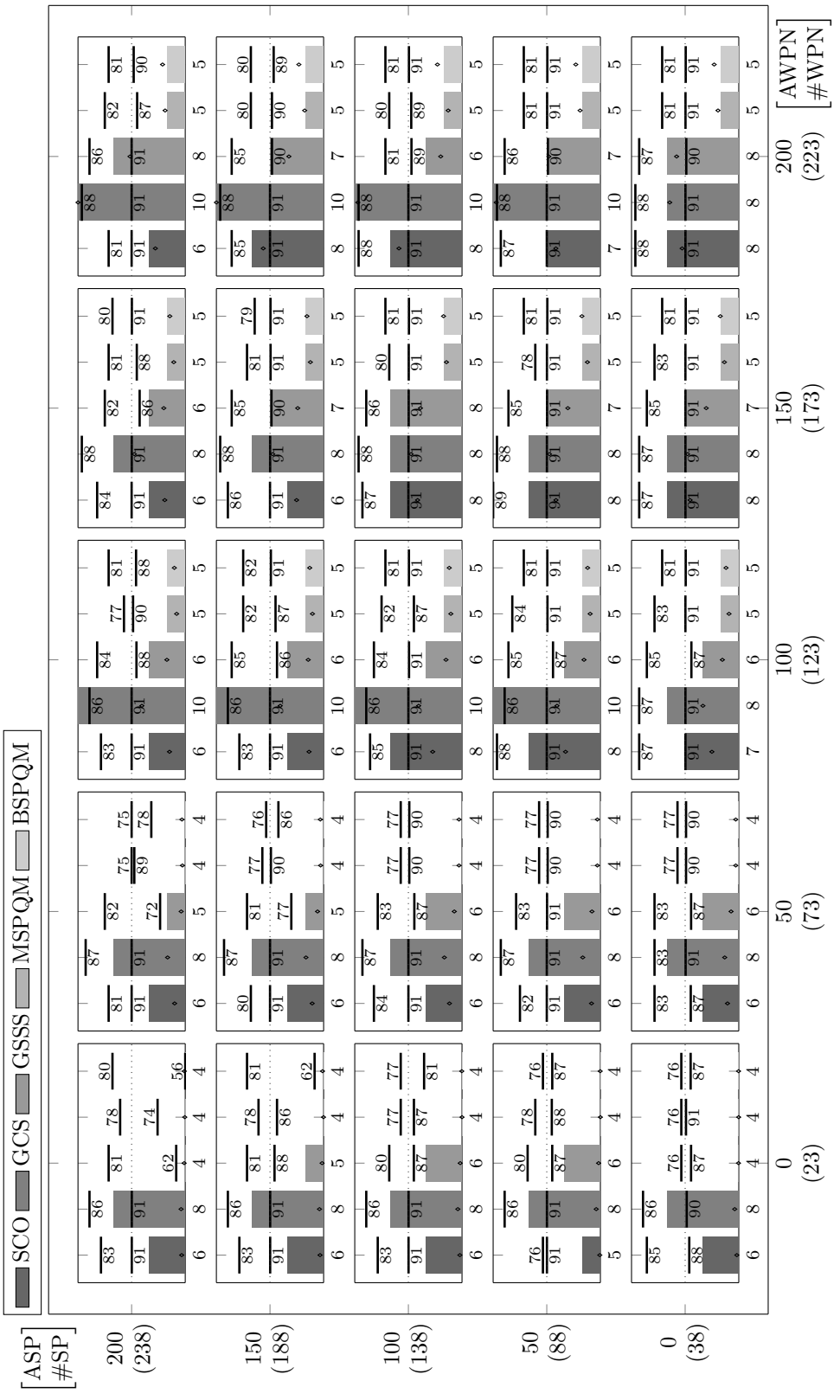


Figure 38: The #SSP, quality, covered area and sum quality for the CR using different levels of SP-WPN discretization.

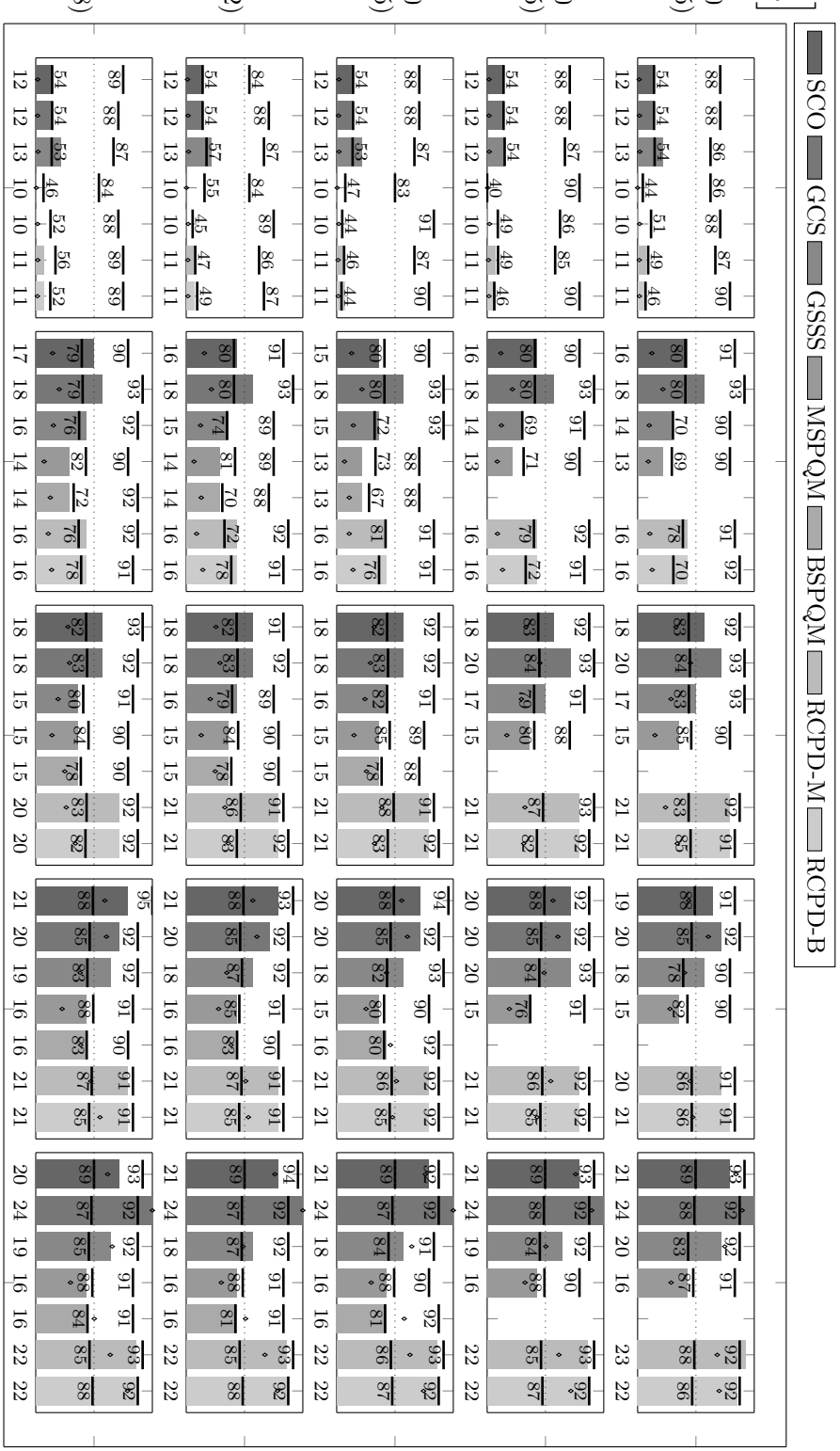


Figure 39: The #SSP, quality, covered area and sum quality for the SF using different levels of SP-WPN discretization.

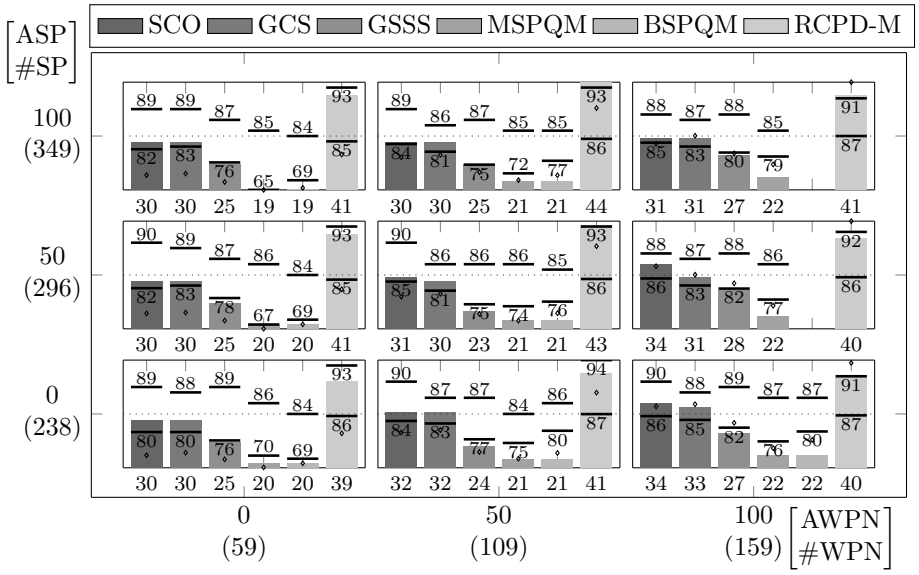


Figure 40: The #SSP, quality, covered area and sum quality for the OF using different levels of SP-WPN discretization.

the #SP is raised above 50. The increase in #SSP for SF and LF is almost linear with an increase in one SPs every 50 WPNs.

Furthermore, the results of the approximate algorithms even show a decrease in #SSP. In the CR results, this behavior can be observed for the SCO and GCS placement results at 100 AWPN. The increase from 0 to 50 ASP results in 10 SSP (GCS) and 8 SSP (SCO), which is a worse overall result than the #SSP selected for 0 ASP (8, 7). For the SF, the same behavior can be observed for example at 100 AWPN and 150 ASP where the approximate algorithms GCS and GSSS result in a worse solution. Similar examples can be found in all environments. Altogether, they show the dependency of the greedy algorithms from the distribution of SPs and WPNs and the chance of the SCO and GCS to select SCs including the same SPs. For the GSSS, the algorithm begins with the best SP candidate in terms of the number of covered WPNs, which means that the distribution of SPs and WPNs has a noticeable influence on the #SSP. Here some more research work on optimized SP and WPN placement might improve its outcome.

In general, it can be seen for all three environments where the optimal MSPQM solutions exist that the number of ASP only results in a maximal decrease of 1 SSP for a constant #WPN. Thus, the initial positions seem to be well chosen as initial input for the models.

Beside the influence on the #SSP, no correlation between the ASP and the mean quality could be observed. A good example is the mean quality of the SF MSPQM solution at 100 AWPN. For up to 50 ASP, it is 90%, then it decreases to 88% (150 AWPN) and finally it is again at 90% for 200 ASP. The same can be observed for the percentage of covered area. In general, when given the mean quality of two solutions

with the same AWPN, the solution with the higher #SSP usually has a slightly higher quality. The behavior of the percentage of the covered area is quite similar. As expected, the overall percentage of covered area is more depended on the AWPN. Especially for the SF, the quality of the MSPQM solution almost doubles from 46% to 82% with the increase from 0 to 50 AWPN.

At last, each environment was evaluated with regard to the geometrical properties of the WPN distribution and SP selection. Due to the extensive number of evaluations, only an excerpt is shown in the appendix (A). Here, for the first three environments CR, SF and LF an extended table of the selected sensors is presented along with the geometrical representation of the best approximate—the GSSS—and the optimal MSPQM solution (if existed) for all combinations of 0, 100 and 200 ASP and AWPN. The OF was not included because only one MSPQM solution exists for it which is already presented in Figure 33.

The visual inspection of the WPN placement for all three environments shows that the additional WPNs are evenly distributed in the environment. Thus, they do seem to minimize the influence on the greedy selection algorithm in terms of forcing SP or SC selections by forming tight clusters in the environment.

The placements of the MSPQM solutions show that SPs at the same position are always well distributed to cover a maximum amount of space with a minimal overlap. Comparing the SF solutions with 0 ASP and 200 AWPN, the upper left rooms shows that an additional sameplace constraint might benefit the GSSS as well. In the lower right entrance of the room, three SPs are selected from which two cover almost the same area and from the MSPQM solution it can be seen that two of them would suffice to cover the whole room. A similar behavior can also be found for the CR solution with 0 ASP and 200 AWPN in the lower right corner.

A comparison of the CR and SF MSPQM solutions with 200 ASP as well as 0 and 200 AWPN shows that the ASP, which are all positions that are not placed at a corner of the environment, are only chosen if the environment contains few AWPN. Thus, for 0 AWPN the SPs are placed closer to the AWPN, whereas the solution with many AWPN forces the selection of SPs at corners. This correlation supports the hypothesis of the initial SP discretization that the corner positions are most valuable for the placement algorithms.

7.6.2 *Discrete Optimizations*

Beside the overall behavior of the mean of max. qualities, the behavior of the BSPQM is of special interest because this algorithm was designed to improve the solution quality. Therefore, it finds the solution with the lowest possible #SSP and the highest sum of all qualities for every WPN. In the bar plots, the sum of all qualities is represented

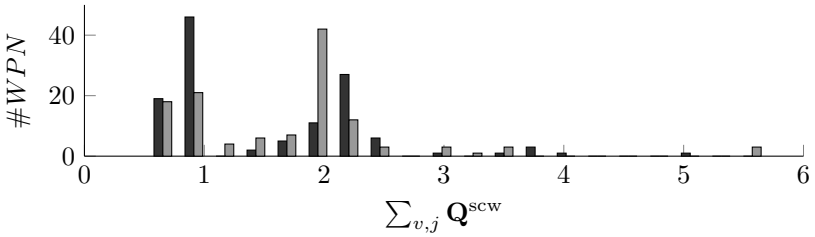


Figure 41: The histogram of the sum of qualities at every WPN for 100 AWPN, 100 ASP for the optimal solution of the MSPQM (dark gray) and BSPQM (light gray) model for the SF. It can be seen that the value distribution, of the BSPQM is shifted towards larger values than the value distribution of the MSPQM.

by the diamond which provides the means to compare different models for this parameter. From the results, it can be seen that the sum of qualities has a strong correlation to the #SSP. Nevertheless, for the CR and SF at 100 AWPN, 100 ASP it can be seen that a higher sum of qualities (101 vs. 99 and 106 vs. 105) does not necessary lead to a higher mean of max. qualities. In detail, the found solution tend to increase the sum of qualities at a few sensor positions. This can be seen in Figure 41 where a histogram over all WPNs of an optimal BSPQM solution is shown for 100 AWPN and 100 ASP. A conclusion that can be derived from this finding is that the BSPQM solutions will not result in a better placement in terms of an evenly distributed quality.

Further, it has to be noted that not every BSPQM solution could be solved to optimality in the given time box. The ones that could not be solved were excluded from the results.

7.6.3 *Approximations*

A comparison of the approximate models for the CR results shows a clear ranking of the three models in which the SCO usually selects less SPs than the GCS and the GSSS outperforms both. Taken all evaluations into account their cross performance can be compared in terms of their "wins", "looses" and "ties" as shown in table 11. The numbers show a clear domination of the GSSS approximation for the SF, LF and OF. Here, only very few of the GCS and SCO solutions provide a better result. Interestingly, for the smaller environments, this number decreases and for the CR the SCO even outperforms the GSSS if both are further improved by the iterative improvement algorithm. The reason is that the iterative improvement works exemplary well on the SCO solutions for the CR, which is also visualized in Figure 42. For almost half of the approximate solutions (1236) SPs could be removed. With only six to eight SPs selected by the SCO for the

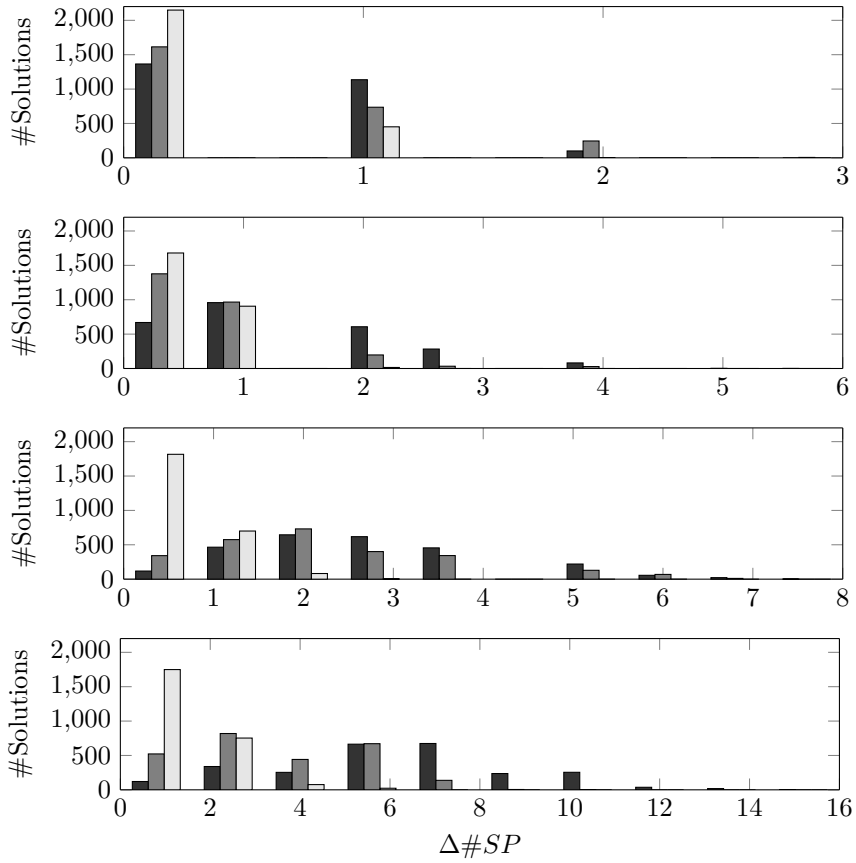


Figure 42: The difference in #SSP as histograms for the CR, SF, LF and OF (top to bottom). The SCO is drawn in dark gray, the GCS in gray and the GSSS in light gray. The ordinate shows the number of solutions and the abscissa the #SSP that were removed by the iterative improvement algorithm.

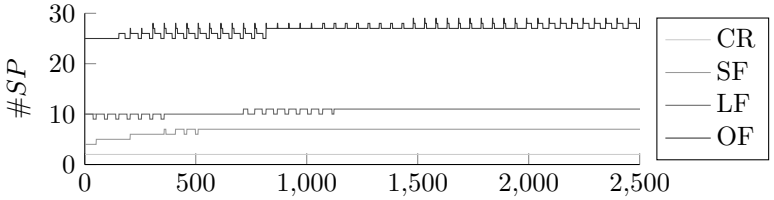


Figure 43: Number of initially selected SPs to cover each WPN within the environment with a single sensor. The abscissa represents the different ASP, AWPN combinations. It can be observed that the CR can be covered with 2 SPs independent of the ASP. The SF can be covered with 4 SPs if there are no AWPN. This number increases to a maximum of 7 SPs for more than 100 AWPN. The LF and OF have a more complicated correlation that depends on the ASP and AWPN combinations. Even though the plot shows the range of necessary SPs for a single coverage.

Env.	GCS vs. SCO			GSSS vs. GCS			GSSS vs. SCO		
	W	L	T	W	L	T	W	L	T
CR	0	2060	541	2493	7	101	1127	711	763
CR (IT)	58	2194	349	2244	44	313	606	885	1110
SF	264	1700	637	2516	18	67	2306	63	232
SF (IT)	112	2162	327	2504	24	73	1870	268	463
LF	602	1409	590	2587	1	13	2596	0	5
LF (IT)	515	1661	425	2587	1	13	2511	28	62
OF	880	1486	235	2589	6	6	2593	2	6
OF (IT)	213	2261	127	2594	2	5	2582	5	14

Table 11: The cross comparison of the three approximation models before and after the iterative improvement (IT), for all four environments. For each environment and comparison, the win (W) column contains the number of SPPs where the first of the two approximations selected less SPs. The loose (L) column contains the number of SPPs with more SPs and the tie (T) column the number of SPPs where both approximations selected the same amount of SPs.

CR, this reduction significantly changes the outcome as shown by the numbers.

An explanation of the good results for the GSSS is the initial single coverage of the environments; it provides a sufficient basis to build a minimum quality two-coverage. Thus, the initially selected SPs can be combined with newly selected sensors to cover each WPN with a minimum quality. In Figure 43 the number of initial SPs is plotted that are needed to cover each environment. Combining the number with the results shown in the bar plots, it can be deduced that the number of finally selected sensors is between two and three times the number of selected sensors for the SF (2.7), LF (2.5), and OF (2.7). The only exception is the CR, where the mean ratio is 3.3.

In Section 5.1.1 a WCAR was derived that can be used to estimate how far off the SCO solution is at worst from an MSPQM solution. Figure 44 shows plots of the WCAR and the real SCO to MSPQM ratio from the experiments. It can be seen that the WCAR underestimates the real solution goodness with an increasing number of #SSP. Additional properties of the problem, like the number of SPs that are visible to each other and may be combined to form sets of unique combinations, need to be taken into account to tighten the bound.

Another finding of the WCAR evaluation is that the real ratio of selected SPs also depends on the environment size. While its mean over all comparable SPP solutions is worst for the CR ($[1.35, 1.69]$) it increases to 1.12 for the iterative improved GSSS at the SF and LF. Thus, for these environments only 12% more SPs are selected in

Environment	SCO	GCS	GSSS
CR	1.42	1.78	1.37
CR (IT)	1.34	1.69	1.35
SF	1.24	1.36	1.14
SF (IT)	1.17	1.28	1.12
LF	1.39	1.37	1.16
LF (IT)	1.25	1.35	1.12

Table 12: Mean of the ratio of the #SP of the greedy solutions and the MSPQM solutions grouped by environment.

average. In detail, the mean of the greedy to MSPQM ratio is given in table 12.

The table also shows the iterative improvement of the approximate solutions (Section 5.1.2) that was applied to every greedy solution to see the improvement in #SSP. Figure 42 shows the difference in #SSP. It can be seen that the GSSS solutions can only be fairly optimized with the proposed improvement with 17%, 35%, 30% and 66% for the CR, SF, LF and OF. In contrast, the percentage of GCS solutions that could be optimized was 37%, 47%, 86% and finally 89% for the OF. The percentage of SCO solutions that could be improved was 43% (CR), 99% (SF) and 100% for the LF and OF. Furthermore, if a solution could be improved, the median number of sensors that were removed ranged from 1 (CR) to 4 (OF) for the GCS and GSSS solutions and 2 (CR) to 9 (OF) for the GSSS solutions.

7.7 SUMMARY

The most important findings of the evaluations are the good performance of the GSSS heuristic, the iterative improvement heuristic and the small influence of the number of ASP on the optimal solution. They show that an input model does not need to have an extensive set of ASP but rather a good discretization in the corners.

In contrast, the results of the WCAR show that it only provides a loose lower bound. Therefore, it should not be used to value approximate solutions in its current form. In addition, the results of the RCPD-based solutions show that the environment separation needs refinement before it can be used for real world setups. Especially the separations within rooms should be used with care because they lead to ill separated polygons.

For an architect of a positioning system, the most important factor to choose an appropriate placement method is the sensor cost in addition to the environment size. If the cost of a sensor is significant, the global optimization models are to be preferred. Does the size of the

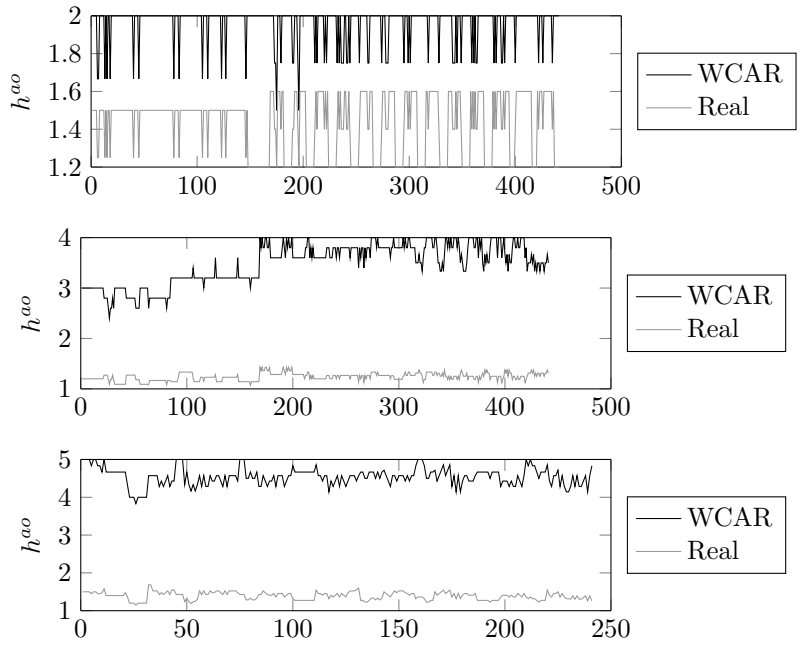


Figure 44: The WCAR and the real ratio for the three environments CR, SF and LF (top to bottom) where the MSPQM and SCO solution could be compared. It can be seen that the CR is the only environment where the ratio of the real solutions meets the WCAR.

environment or the discretization exactness lead to insolvable model sizes, a combination of the GSSS and the iterative improvement, along with a separate search for sensor overlappings is the suggested procedure. Both methods may further be combined with the continuous methods to improve the quality of coverage.

8

CONCLUSION AND OUTLOOK

In this thesis, it was shown how sensor placement problems (SPPs) can be stated and solved. New techniques to model such problems and to find approximate solutions were presented. Their evaluation was conducted for real world environments using the properties of a self-build positioning system.

Nine methods are presented to search for an optimal sensor placement in a discrete environment, two global optimization models, two greedy heuristics, two nonlinear optimization models, two decomposition based optimization models and one approximate optimization model. All of these methods serve the same goal, to minimize the number of sensors while serving the positioning constraints; a minimum quality based on the sensor geometry. Two of the methods also introduce secondary goals that become active if the primary goal is fulfilled. The Best Sensor Pairwise Quality Model (BSPQM) optimize the sum of qualities over all workspace positions (WPNs) and the Continuous Maximum Quality Model (CMQM) increases the area that is covered with a minimum quality.

The global optimization models BSPQM and Minimum Sensor Pairwise Quality Model (MSPQM) solve the full problem. The approximate optimization model Sensor Combination Optimization (SCO) and the greedy models Greedy Single Sensor Selection (GSSS) and Greedy Combined Selection (GCS) solve an approximation of the problem. The decomposition based optimization models RCPD combined with BSPQM (RCPD-B) and RCPD combined with MSPQM (RCPD-M) separate the input environment into multiple small models. Therefore, their solution to the SPP is a combination of all independently calculated solutions. To state the decomposition based models, a convex polygon decomposition algorithm is introduced. It has a polynomial complexity and separates an input polygon into convex pieces with additional Steiner points that are only created on the polygon boundary.

A problem with the approximation algorithms is the estimation of their solution quality. To better estimate the solution quality, the means of computing the worst case approximation ratio (WCAR) for the SCO was shown. The derivation also introduced properties of approximate SPP solutions that can be exploited to improve them. This lead to a problem specific search heuristic, which can be used to iteratively improve approximate solutions.

All algorithms were validated using the digitalization of four real world input environments of different size. Each environment was

solved with an extensive set of varying discretization exactness, leading to more than 15,000 solved or improved SPP for each environment. The most important findings of the evaluation are the good performance of the GSSS algorithm over the other approximation algorithms and the negligible influence of the sensor pose (SP) discretization exactness for the smaller environments. In addition, it could be shown that the WCAR tends to be too pessimistic for larger problems. At last, it was shown that the iterative improvement of approximate solutions can significantly increase the solution goodness.

Altogether, the presented work is intended to serve as a guide for engineers of positioning systems that need to solve similar problems. Nevertheless, its finite scope leaves some room for improvements. In particular, the polygon decomposition algorithm might be combined with a more sophisticated method to solve the partial problems for instance by populating found solutions as suggested in the iterative positioning scheme. In addition, the discretization schemes should serve well for an iterative refinement process in which an optimal solution is computed for a small input set and then used as an initial solution for a refined discretization. Further, the greedy methods might serve as an initial guess for the optimization process to improve the solving time.

In turn, using additional sameplace constraints in the approximate models and combining them with the iterative improvement might be a valid method to further close the gap between the global optimal and the approximate solutions.

BIBLIOGRAPHY

Bai, X., Kumar, S., Xuan, D., Yun, Z., & Lai, T. (2006). Deploying wireless sensors to achieve both coverage and connectivity. In *Proceedings of the 7th acm international symposium on mobile ad hoc networking and computing* (pp. 131–142). New York, New York, USA: ACM Press

Balakrishnan, H., Supervisor, T., & Smith, A. C. (2005). *The Cricket Indoor Location System* (Doctoral Thesis, Massachusetts Institute of Technology).

Beutel, J., Kasten, O., & Ringwald, M. (2003). Poster Abstract: BTnodes – A Distributed Platform for Sensor Nodes. *1st International Conference on Embedded Networked Sensor Systems*, 292–293

Bradley, S. P., Hax, A. C., & Magnanti, T. L. (1977). *Mathematical Programming: An Overview*. New York: Addison-Wesley.

Cai, Y., Lou, W., Li, M., & Li, X. Y. (2009). Energy efficient target-oriented scheduling in directional sensor networks. *IEEE Transactions on Computers*, *58*(9), 1259–1274

Chazelle, B. (1991). Triangulating a simple polygon in linear time. *Discrete & Computational Geometry*, *6*(1), 485–524

Chazelle, B. & Palios, L. (1994). Decomposition Algorithms in Geometry. *Algebraic Geometry and its Applications*, 419–447

Chong, C.-Y. & Kumar, S. P. (2003). Sensor networks: evolution, opportunities, and challenges. *Proceedings of the IEEE*, *91*(8).

Conci, N. & Lizzi, L. (2009). Camera placement using particle swarm optimization in visual surveillance applications. *2009 16th IEEE International Conference on Image Processing (ICIP)*, 3485–3488

Contreras, D., Castro, M., & Torre, D. S. D. L. (2014). Performance evaluation of bluetooth low energy in indoor positioning systems. *European Transactions on Telecommunications*, *25*(October), 294–307. arXiv: arXiv:1307.8198v1

de Berg, M., van Kreveld, M., & Overmars, M. (2008). *Computational Geometry: Algorithms and Applications* (third). Springer

Deak, G., Curran, K., & Condell, J. (2012). A survey of active and passive indoor localisation systems. *Computer Communications*, *35*(16), 1939–1954.

ElGindy, H., Everett, H., & Toussaint, G. (1993). Slicing an ear using prune-and-search. *Pattern Recognition Letters*, *14*(9), 719–722.

Erdem, U. M. & Sclaroff, S. (2004). Optimal placement of cameras in floorplans to satisfy task requirements and cost constraints. *Proceedings of the International Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras*, 30–41

Erdem, U. M. & Sclaroff, S. (2006). Automated camera layout to satisfy task-specific and floor plan-specific coverage requirements. *Computer Vision and Image Understanding*, *103*(3), 156–169

Fallah, N., Apostolopoulos, I., Bekris, K., & Folmer, E. (2013). Indoor human navigation systems: A survey. *Interacting with Computers*, *25*(1), 21–33

Finkel, R. a. & Bentley, J. L. (1974). Quad trees a data structure for retrieval on composite keys. *Acta Informatica*, *4*(1), 1–9.

Franklin, W. R. (1989). *Art Gallery Theorems and Algorithms (Joseph O'Rourke)*. Oxford: Oxford University Press

Ghosh, S. K. (2007). *Visibility Algorithms in the Plane*. Cambridge: Cambridge University Press

Goodman, J. E. & O'Rourke, J. (2004). *Handbook of Discrete and Computational Geometry*. Boca Raton: Chapman & Hall/CRC

Greiner, G. & Hormann, K. (1998). Efficient clipping of arbitrary polygons. *ACM Transactions on Graphics*, *17*(2), 71–83

Han, X., Cao, X., Lloyd, E. L., & Shen, C. C. (2008). Deploying directional sensor networks with guaranteed connectivity and coverage. *5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, 153–160

Hansen, N. & Ostermeier, a. (2001). Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, *9*(2), 159–195

Hauschildt, D., Kemper, J., Kirchhof, N., Juretko, B., & Linde, H. (2010). Real-time scene simulator for thermal infrared localization. In *Proceedings - winter simulation conference* (pp. 879–890). IEEE

Hauschildt, D. & Kirchhof, N. (2010). Advances in thermal infrared localization: Challenges and solutions. In *International conference on indoor positioning and indoor navigation* (September, pp. 1–8). IEEE

Hauschildt, D. & Kirchhof, N. (2011). Improving indoor position estimation by combining active TDOA ultrasound and passive thermal infrared localization. In *Proceedings of the 8th workshop on positioning navigation and communication* (pp. 94–99). IEEE

Hertel, S. & Mehlhorn, K. (1985). Fast triangulation of the plane with respect to simple polygons. *Information and Control*, *64*(1-3), 52–76

Hightower, J. & Borriello, G. (2001). A Survey and Taxonomy of Location Systems for Ubiquitous Computing. *IEEE Computer*, *34*(01-08-03), 57–66

Honsberger, R. (1976). *Mathematical Gems II*. Mathematical Assn of Amer.

Hörster, E. & Lienhart, R. (2006a). Approximating optimal visual sensor placement. In *Ieee international conference on multimedia and expo* (Vol. 2006, pp. 1257–1260). IEEE

Hörster, E. & Lienhart, R. (2006b). *On the optimal placement of multiple visual sensors* (tech. rep. No. 3). Institut fuer Informatik. Augsburg: ACM Press

Hromkovič, J. (2004). *Algorithmics for Hard Problems*. Texts in Theoretical Computer Science. An EATCS Series. Berlin, Heidelberg: Springer Berlin Heidelberg.

IBM. (2015). IBM CPLEX Optimizer. Retrieved June 24, 2015, from http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/index.html

Iozzio, C. (2014). Indoor Mapping Lets the Blind Navigate Airports. Retrieved May 24, 2015, from http://www.smithsonianmag.com/innovation/indoor-mapping-lets-blind-navigate-airports-180952292/?no-ist

Isler, V. & Bajksy, R. (2006). The sensor selection problem for bounded uncertainty models. *IEEE Transactions on Automation Science and Engineering*, *3*(4), 372–381.

Kallrath, J. (2013). *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*. Wiesbaden: Springer Fachmedien Wiesbaden

Karp, R. (1972). Reducibility among Combinatorial Problems. *Complexity of Computer Computations*, (Chapter 9), 85–103

Keil, J. (2000). Polygon decomposition. *Handbook of Computational Geometry*, 491–518.

Kelly, A. (2003). Precision dilution in triangulation based mobile robot position estimation. In *In intelligent autonomous systems*. Amsterdam.

Kemper, J. (2010). *Passive Infrarot-Lokalisierung* (Doctoral dissertation, TU Dortmund).

Kemper, J. & Hauschildt, D. (2010). Passive infrared localization with a probability hypothesis density filter. In *Proceedings of the 7th workshop on positioning, navigation and communication* (pp. 68–76). IEEE

Kemper, J., Walter, M., & Linde, H. (2008). Human-assisted calibration of an angulation based indoor location system. *Second International Conference on Sensor Technologies and Applications*, *3*(1&2), 196–201

Kim, K. (2007). *Spatial analytical approaches for supporting security monitoring* (PhD Thesis, Ohio State University).

Kirchhof, N. (2013). Optimal placement of multiple sensors for localization applications. In *International conference on indoor positioning and indoor navigation* (October, p. 10).

Kivimäki, T., Vuorela, T., Peltola, P., & Vanhala, J. (2014). A review on device-free passive indoor positioning methods. *International Journal of Smart Home*, *8*(1), 71–94.

Kjærgaard, M. B., Blunck, H., Godsk, T., Toftkjær, T., Christensen, D. L., & Grønbæk, K. (2010). Indoor Positioning Using GPS Re-

visited. *Proceedings of the 8th international conference on Pervasive Computing*, *6030*, 38–56

Korte, B. & Vygen, J. (2010). *Combinatorial Optimization*. Algorithms and Combinatorics. Berlin, Heidelberg: Springer Berlin Heidelberg

Kouakou, M., Yamamoto, S., & Yasumoto, K. (2010). Cost-Efficient Deployment for Full-Coverage and Connectivity in Indoor 3D WSNs. *Proceedings of the 12th ACM international conference adjunct papers on Ubiquitous computing*.

Kumar, S., Lai, T. H., & Balogh, J. (2008). On k-coverage in a mostly sleeping sensor network. *Wireless Networks*, *14*(3), 277–294

Laboratory, L. (1986). *Distributed Sensor Networks*. Massachusetts Institute of Technology. Lexinton, Massachusetts.

Lang, T. J. & Adams, W. S. (1998). A Comparison of Satellite Constellations for Continuous Global Coverage. In J. Ha (Ed.), *Mission design & implementation of satellite constellations* (pp. 51–62). Springer Netherlands.

Längkvist, M., Karlsson, L., & Loutfi, A. (2014). A review of unsupervised feature learning and deep learning for time-series modelling. *Pattern Recognition Letters*.

Lien, J. M. & Amato, N. M. (2008). Approximate convex decomposition of polyhedra and its applications. *Computer Aided Geometric Design*, *25*(7), 503–522

Liu, L., Zhang, X., & Ma, H. (2011). Localization-oriented coverage in wireless camera sensor networks. *IEEE Transactions on Wireless Communications*, *10*(2), 484–494

Lohr, S. (2009). *Sampling: Design and Analysis*. Boston, MA: Cengage Learning.

Lymberopoulos, D., Liu, J., Yang, X., Choudhury, R. R., Handziski, V., & Sen, S. (2015). A realistic evaluation and comparison of indoor location technologies. In *Proceedings of the 14th international conference on information processing in sensor networks - ipsn '15* (Table 1, pp. 178–189). New York, New York, USA: ACM Press

Mautz, R. (2012). *Indoor Positioning Technologies* (Doctoral dissertation, ETH Zurich)

Mini, S., Udgata, S. K., & Sabat, S. L. (2012). M-Connected Coverage Problem in Wireless Sensor Networks. *ISRN Sensor Networks*, *2012*, 1–9

Mittal, A. & Davis, L. S. (2004). Visibility Analysis and Sensor Planning in Dynamic Environments. In *European conference on computer vision 2004* (pp. 175–189)

Mittal, A. & Davis, L. S. (2007). A General Method for Sensor Planning in Multi-Sensor Systems: Extension to Random Occlusion. *International Journal of Computer Vision*, *76*(1), 31–52

Nam, Y. & Hong, S. (2012). Optimal placement of multiple visual sensors using simulation of pedestrian movement. *International Conference on Computing, Networking and Communications*, 67–71

Osais, Y., St-Hilaire, M., & Yu, F. (2009). On Sensor Placement for Directional Wireless Sensor Networks. *2009 IEEE International Conference on Communications*, 1–5

Potort, F., Barsocchi, P., & Girolami, M. (2015). Evaluating indoor localization solutions in large environments through competitive benchmarking : the EvAAL-ETRI Competition. *Proceedings of the Sixth International Conference on Indoor Positioning and Indoor Navigation*, (October), 13–16.

Rosen, K. (2011). *Discrete Mathematics and Its Applications* (4th editio). William C Brown Pub.

Sarker, R. & Newton, C. (2007). *Optimization Modelling*. CRC Press

Schirra, S. & Schirra, S. (2000). *Robustness and precision issues in geometric computation* (tech. rep. No. 21957).

Shewchuk, J. R. (1997). Adaptive Precision Floating-Point Arithmetic and Fast Robust Geometric Predicates. *Discrete & Computational Geometry*, *18*(3), 305–363

Shimrat, M. (1962). Algorithm 112: Position of point relative to polygon. *Communications of the ACM*, *5*(8), 434

Song, B., Ding, C., Kamal, A., Farrell, J., & Roy-chowdhury, A. (2011). Distributed Camera Networks. *IEEE Signal Processing Magazine*, *28*(3), 20–31

Swanson, E. R. (1978). Geometric Dilution of Precision. *Navigation*, *25*(4), 425–429

Tarabanis, K. a., Allen, P. K., & Tsai, R. Y. (1995). Survey of sensor planning in computer vision. *IEEE Transactions on Robotics and Automation*, *11*(1), 86–104

Tekdas, O. & Isler, V. (2010). Sensor placement for triangulation-based localization. *IEEE Transactions on Automation Science and Engineering*, *7*(3), 681–685.

Thrun, S. (2002). Robotic mapping: A survey. *Exploring artificial intelligence in the new millennium*, (February).

Torrieri, D. (1984). Statistical Theory of Passive Location Systems. *IEEE Transactions on Aerospace and Electronic Systems*, *AES-20*(2), 183–198

Ubisense. (2015). Ubisense boosts productivity at BMW. Retrieved May 24, 2015, from http://www.ubisense.net/en/information/resources/automotive-rework

Vereinigung der Metall-Berufsgenossenschaften. (2001). *BGI 523 Mensch und Arbeitsplatz*. Vereinigung der Metall-Berufsgenossenschaften.

Walker, J. (1971). *Some Circular Orbit Patterns Providing Continuous Whole Earth Coverage*. DTIC Document.

Watkins, P. R. (1990). *Integer and Combinatorial Optimization*. New Jersey: John Wiley & Sons

Witte, B. & Sparla, P. (2011). *Vermessungskunde und Grundlagen der Statistik für das Bauwesen*. Heidelberg: Wichmann.

Yao, Y., Chen, C. H., Abidi, B., Page, D., Koschan, A., & Abidi, M. (2008). Sensor planning for automated and persistent object tracking with multiple cameras. *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 1–8

Yelbay, B., Birbil, Ş. İ., & Bülbül, K. (2014). The set covering problem revisited: An empirical study of the value of dual information. *Journal of Industrial and Management Optimization*, *11*(2), 575–594

Zhang, L., Tang, J., & Zhang, W. (2009). Strong barrier coverage with directional sensors. *IEEE Global Telecommunications Conference*, 1–6

Zhao, X., Xiao, Z., Markham, A., Trigoni, N., & Ren, Y. (2014). Does BTLE measure up against WiFi? A comparison of indoor location performance. *20th European Wireless Conference*, 1–6.

Zhu, J. (1992). Calculation of geometric dilution of precision. *IEEE Transactions on Aerospace and Electronic Systems*, *28*(3), 893–894

A

EXTENDED EVALUATION RESULTS

An Extended presentation of experimental results.

A.1 CONFERENCE ROOM

A.1.1 *Statistics*

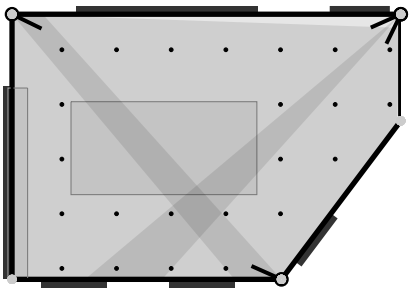
A.1.1.1 *Number of Selected Sensors*

Env	A	B	C	D	E	F	G	H	I	J
(38, 23)	—	—	8	8	4	4	4	4	4	4
(38, 123)	—	—	8	8	6	5	5	5	5	4
(38, 223)	—	—	8	8	8	5	5	5	5	4
(38, 323)	—	—	10	10	7	—	—	—	—	4
(38, 423)	—	—	10	8	7	—	—	—	—	4
(38, 523)	—	—	9	8	7	—	—	—	—	4
(138, 23)	—	—	8	8	6	4	4	4	4	4
(138, 123)	—	—	10	10	6	5	5	5	5	4
(138, 223)	—	—	10	10	6	5	5	5	5	4
(138, 323)	—	—	10	10	6	—	—	—	—	4
(138, 423)	—	—	10	8	7	—	—	—	—	4
(138, 523)	—	—	10	8	7	—	—	—	—	4
(238, 23)	—	—	8	8	4	4	4	4	4	3
(238, 123)	—	—	10	10	6	5	5	5	5	4
(238, 223)	—	—	10	10	8	5	5	5	5	4
(238, 323)	—	—	10	10	7	—	—	—	—	4
(238, 423)	—	—	10	10	7	—	—	—	—	4
(238, 523)	—	—	9	8	7	—	—	—	—	4
(338, 23)	—	—	6	6	4	—	—	—	—	3
(338, 123)	—	—	8	8	6	—	—	—	—	4
(338, 223)	—	—	7	8	7	—	—	—	—	4
(338, 323)	—	—	10	8	7	—	—	—	—	4
(338, 423)	—	—	10	10	7	—	—	—	—	4

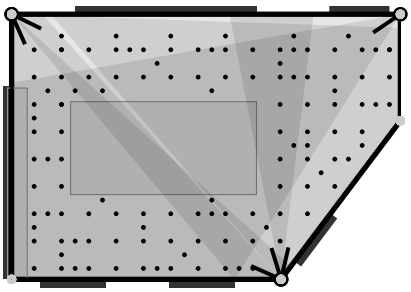
Env	A	B	C	D	E	F	G	H	I	J
(338, 523)	—	—	9	8	7	—	—	—	—	4
(438, 23)	—	—	6	6	4	—	—	—	—	3
(438, 123)	—	—	8	8	6	—	—	—	—	4
(438, 223)	—	—	7	8	6	—	—	—	—	4
(438, 323)	—	—	8	8	6	—	—	—	—	4
(438, 423)	—	—	8	8	7	—	—	—	—	4
(438, 523)	—	—	9	10	7	—	—	—	—	4
(538, 23)	6	4	6	6	4	—	—	—	—	3
(538, 123)	8	4	8	8	7	—	—	—	—	4
(538, 223)	7	4	7	8	7	—	—	—	—	4
(538, 323)	8	4	8	8	7	—	—	—	—	4
(538, 423)	8	5	8	8	7	—	—	—	—	4
(538, 523)	9	5	9	10	7	—	—	—	—	4

A.1.2 *Placements*

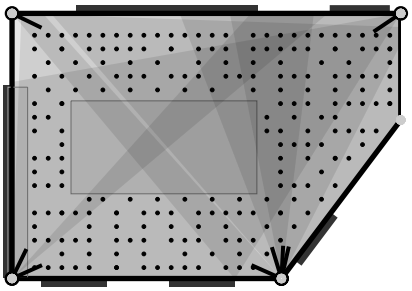
A.1.2.1 *GSSS*



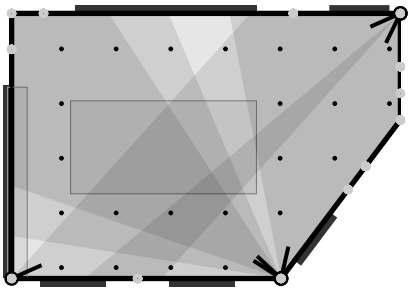
(0, 0, 38, 4, 76%, 87%)



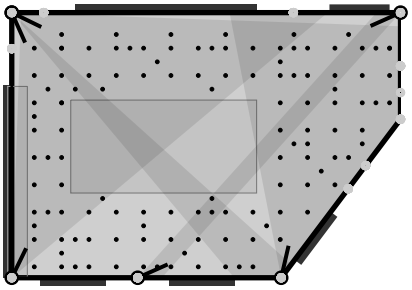
(0, 100, 38, 6, 85%, 87%)



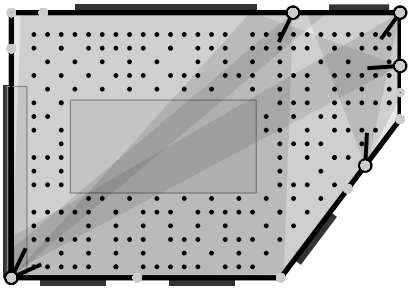
(0, 200, 38, 8, 87%, 90%)



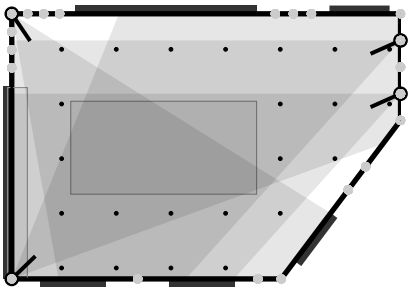
(100, 0, 138, 6, 80%, 87%)



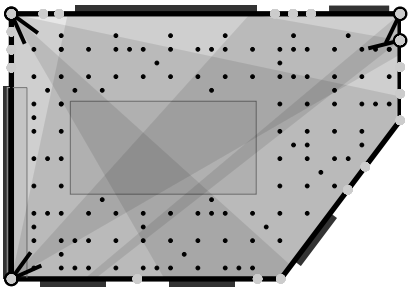
(100, 100, 138, 6, 84%, 91%)



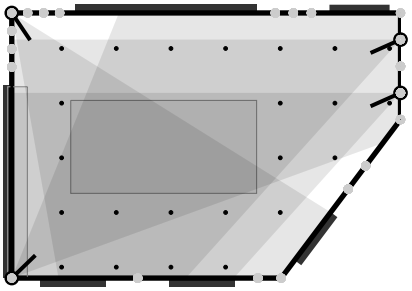
(100, 200, 138, 6, 81%, 89%)



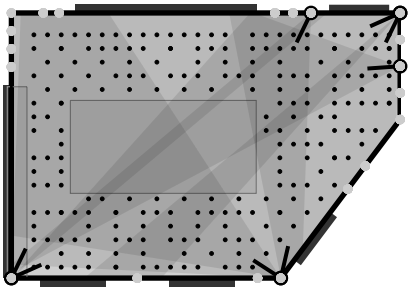
(200, 0, 238, 4, 81%, 62%)



(200, 100, 238, 6, 84%, 88%)

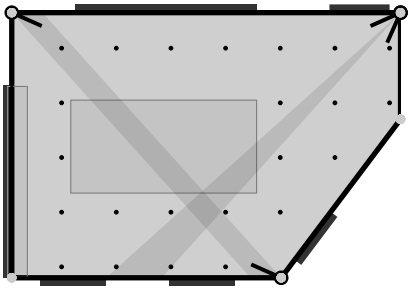


(200, 0, 238, 4, 81%, 62%)

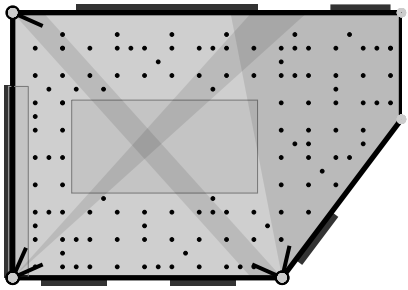


(200, 200, 238, 8, 86%, 91%)

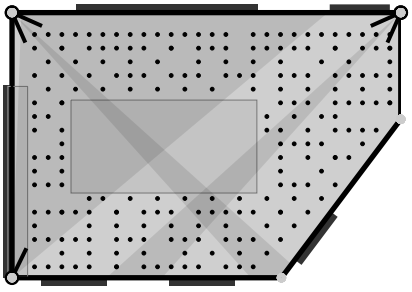
A.1.2.2 *MSPQM*



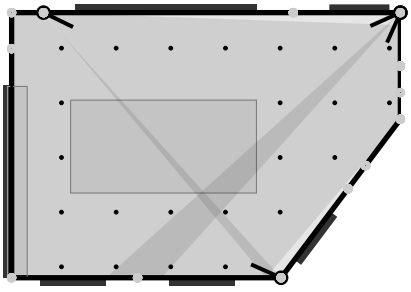
(0, 0, 38, 4, 76%, 90%)



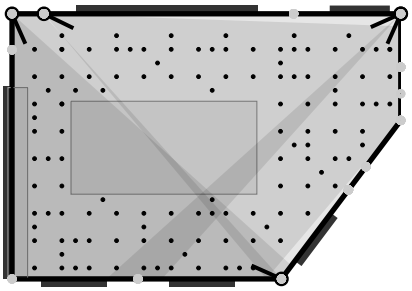
(0, 100, 38, 5, 83%, 91%)



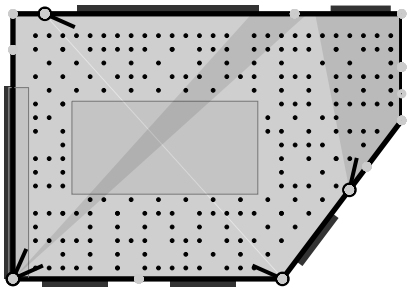
(0, 200, 38, 5, 81%, 91%)



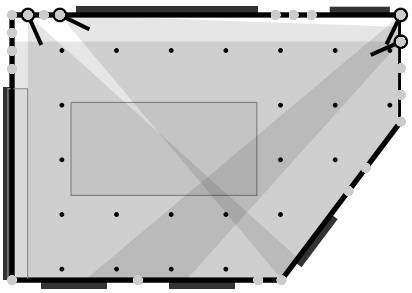
(100, 0, 138, 4, 77%, 87%)



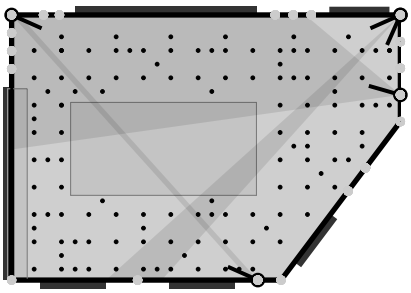
(100, 100, 138, 5, 82%, 87%)



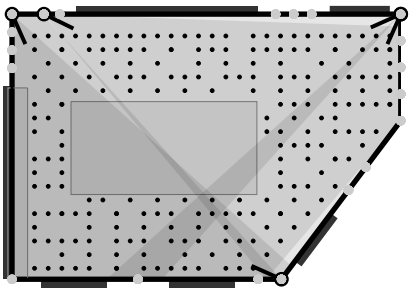
(100, 200, 138, 5, 80%, 89%)



(200, 0, 238, 4, 78%, 74%)



(200, 100, 238, 5, 77%, 90%)



(200, 200, 238, 5, 82%, 87%)

A.2 SMALL FLAT

A.2.1 *Statistics*

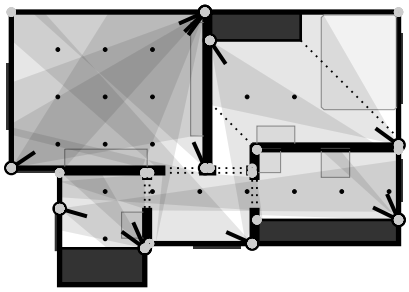
A.2.1.1 *Number of Selected Sensors*

Env	A	B	C	D	E	F	G	H	I	J
(118, 19)	—	—	12	12	13	10	10	11	11	9
(135, 119)	—	—	22	18	15	15	15	20	20	14
(135, 219)	—	—	24	24	19	16	16	22	22	14
(135, 319)	—	—	26	24	20	—	—	22	—	14
(135, 419)	—	—	28	26	22	—	—	24	—	15
(135, 519)	—	—	30	24	22	—	—	25	—	15
(218, 19)	—	—	12	12	13	10	10	11	11	9
(235, 119)	—	—	22	18	16	15	15	21	21	14
(235, 219)	—	—	23	24	18	16	16	22	22	14
(235, 319)	—	—	26	24	21	—	—	22	—	14
(235, 419)	—	—	29	26	21	—	—	24	—	15
(235, 519)	—	—	30	26	21	—	—	23	—	15
(318, 19)	—	—	12	12	13	10	10	11	11	9
(335, 119)	—	—	23	20	17	15	—	21	21	14
(335, 219)	—	—	24	24	20	16	—	23	22	14
(335, 319)	—	—	27	24	20	—	—	22	—	14
(335, 419)	—	—	28	26	22	—	—	23	—	15
(335, 519)	—	—	28	26	21	—	—	23	—	15
(418, 19)	—	—	12	12	13	—	—	11	11	9
(435, 119)	—	—	23	20	18	—	—	21	—	13
(435, 219)	—	—	22	24	20	—	—	22	—	14
(435, 319)	—	—	29	24	19	—	—	22	—	14
(435, 419)	—	—	29	24	19	—	—	24	—	14
(435, 519)	—	—	27	24	20	—	—	23	—	14
(518, 19)	—	—	12	12	11	—	—	11	11	9
(535, 119)	—	—	23	20	16	—	—	20	—	13
(535, 219)	—	—	22	24	19	—	—	23	—	14
(535, 319)	—	—	29	24	20	—	—	22	—	14
(535, 419)	—	—	28	24	21	—	—	24	—	14
(535, 519)	—	—	27	24	20	—	—	23	—	14
(618, 19)	12	—	12	12	11	—	—	11	11	9
(635, 119)	19	20	21	18	17	—	—	19	—	12
(635, 219)	21	21	21	22	19	—	—	23	—	13

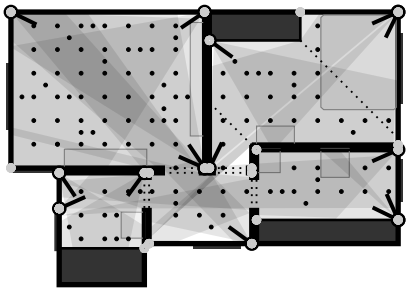
Env	A	B	C	D	E	F	G	H	I	J
(635, 319)	29	26	29	24	18	—	—	22	—	14
(635, 419)	28	27	28	24	21	—	—	23	—	14
(635, 519)	26	26	27	24	21	—	—	24	—	14

A.2.2 *Placements*

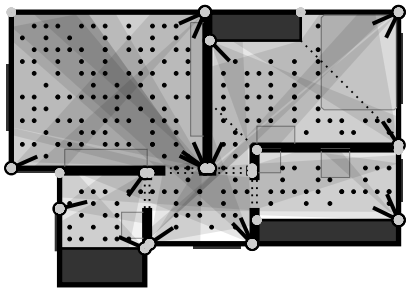
A.2.2.1 *GSSS*



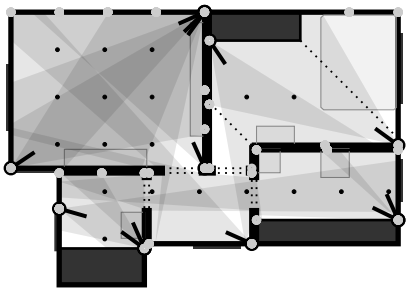
(0, 0, 118, 13, 87%, 53%)



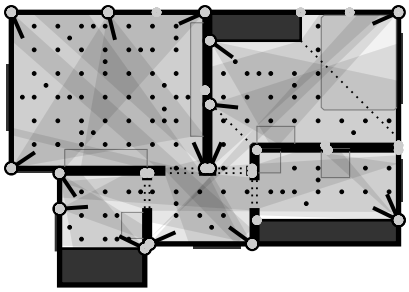
(0, 100, 135, 15, 91%, 80%)



(0, 200, 135, 19, 92%, 85%)

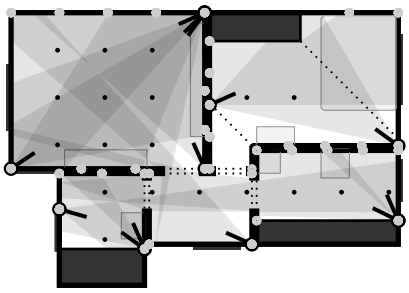


(100, 0, 218, 13, 87%, 53%)

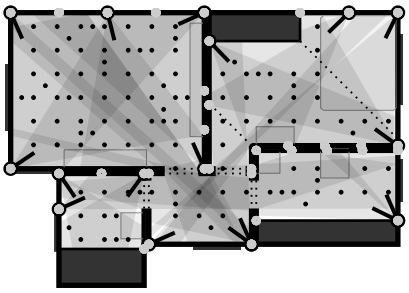


(100, 100, 235, 16, 91%, 82%)

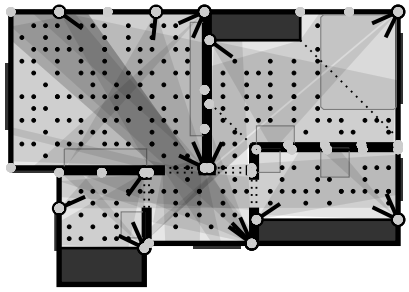
(100, 200, 235, 18, 91%, 84%)



(200, 0, 318, 13, 86%, 54%)

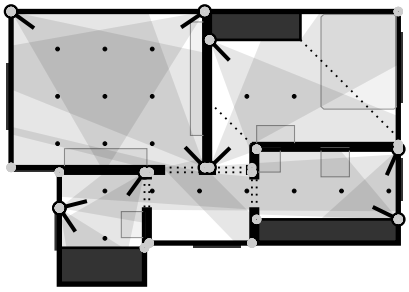


(200, 100, 335, 17, 93%, 83%)

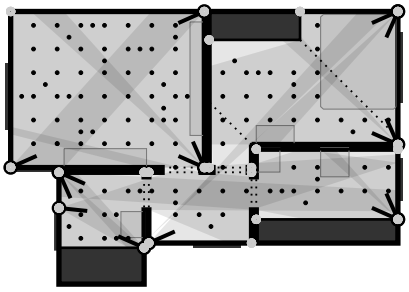


(200, 200, 335, 20, 92%, 82%)

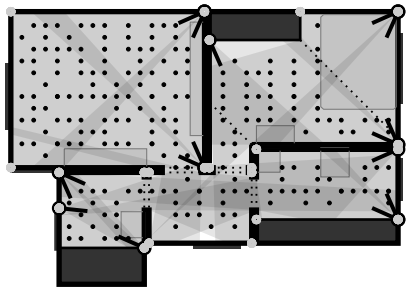
A.2.2.2 *MSPQM*



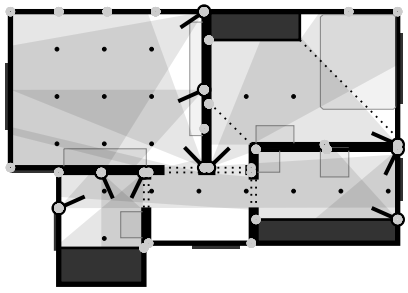
(0, 0, 118, 10, 84%, 46%)



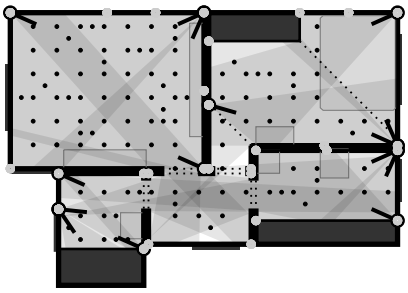
(0, 100, 135, 15, 90%, 84%)



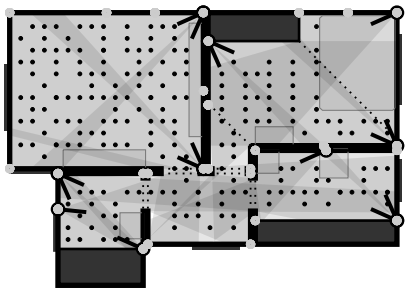
(0, 200, 135, 16, 91%, 88%)



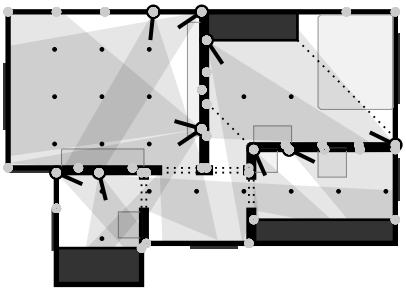
(100, 0, 218, 10, 83%, 47%)



(100, 100, 235, 15, 89%, 85%)

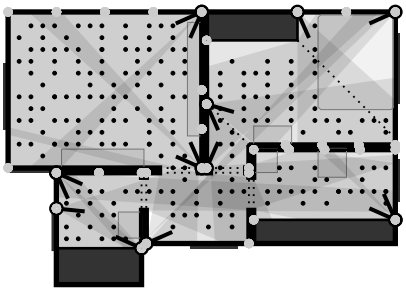


(100, 200, 235, 16, 90%, 88%)



(200, 0, 318, 10, 86%, 44%)

(200, 100, 335, 15, 90%, 85%)



(200, 200, 335, 16, 91%, 87%)

A.3 LARGE FLAT

A.3.1 *Statistics*

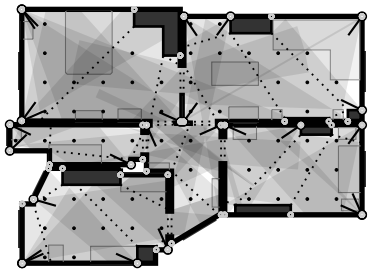
A.3.1.1 *Number of Selected Sensors*

Env	A	B	C	D	E	F	G	H	I	J
(238, 59)	—	—	31	30	25	20	20	39	43	19
(249, 159)	—	—	36	33	27	22	22	40	45	21
(251, 259)	—	—	43	36	28	—	—	49	53	22
(251, 359)	—	—	44	38	28	—	—	48	—	23
(251, 459)	—	—	42	38	27	—	—	49	—	23
(251, 559)	—	—	48	39	26	—	—	51	—	23
(338, 59)	—	—	32	30	25	19	19	41	44	19

Env	A	B	C	D	E	F	G	H	I	J
(349, 159)	—	—	38	31	27	22	—	41	47	21
(351, 259)	—	—	44	36	27	23	—	51	52	22
(351, 359)	—	—	43	38	28	—	—	51	—	23
(351, 459)	—	—	41	38	29	—	—	53	—	23
(351, 559)	—	—	48	40	27	—	—	56	—	23
(438, 59)	—	—	32	30	24	—	—	40	44	18
(449, 159)	—	—	35	32	23	20	—	43	46	19
(451, 259)	—	—	45	38	27	23	—	51	55	21
(451, 359)	—	—	42	36	28	—	—	53	—	22
(451, 459)	—	—	42	38	26	—	—	55	—	22
(451, 559)	—	—	43	38	30	—	—	55	—	22
(538, 59)	—	—	31	28	24	—	—	40	—	18
(549, 159)	—	—	34	30	27	—	—	42	—	19
(551, 259)	—	—	42	38	29	—	—	49	—	21
(551, 359)	—	—	40	36	28	—	—	48	—	22
(551, 459)	—	—	42	38	29	—	—	53	—	22
(551, 559)	—	—	40	36	30	—	—	53	—	22
(638, 59)	—	—	29	26	24	—	—	36	—	18
(649, 159)	—	—	32	32	24	—	—	42	—	19
(651, 259)	—	—	42	36	30	—	—	48	—	21
(651, 359)	—	—	40	36	26	—	—	50	—	21
(651, 459)	—	—	39	36	30	—	—	54	—	21
(651, 559)	—	—	40	36	30	—	—	54	—	21
(738, 59)	28	28	29	26	23	—	—	38	—	18
(749, 159)	29	29	32	30	23	—	—	43	—	19
(751, 259)	42	42	42	36	27	—	—	49	—	21
(751, 359)	39	39	40	36	30	—	—	51	—	21
(751, 459)	39	39	39	36	28	—	—	53	—	21
(751, 559)	39	39	40	36	30	—	—	54	—	21

A.3.2 *Placements*

A.3.2.1 *GSSS*



(0, 0, 238, 25, 89%, 76%)

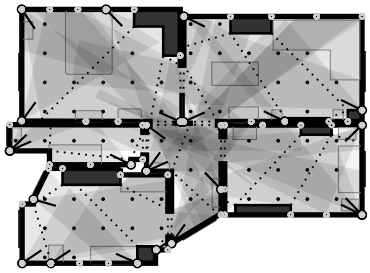
(0, 100, 249, 27, 89%, 82%)

(0, 200, 251, 28, 89%, 85%)

(100, 0, 338, 25, 87%, 76%)

(100, 100, 349, 27, 88%, 80%)

(100, 200, 351, 27, 89%, 85%)



(200, 0, 438, 24, 86%, 78%)

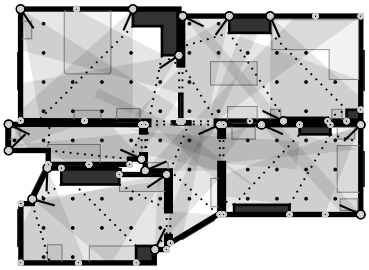
(200, 100, 449, 23, 85%, 79%)

(200, 200, 451, 27, 90%, 85%)

A.3.2.2 *MSPQM*

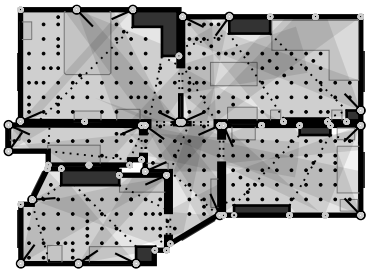
(0, 0, 238, 20, 86%, 70%)

(0, 100, 249, 22, 87%, 76%)



(100, 0, 338, 19, 85%, 65%)

(100, 100, 349, 22, 85%, 79%)



(100, 200, 351, 23, 87%, 82%)

(200, 100, 449, 20, 85%, 76%)

(200, 200, 451, 23, 88%, 82%)

A.4 OFFICE FLOOR

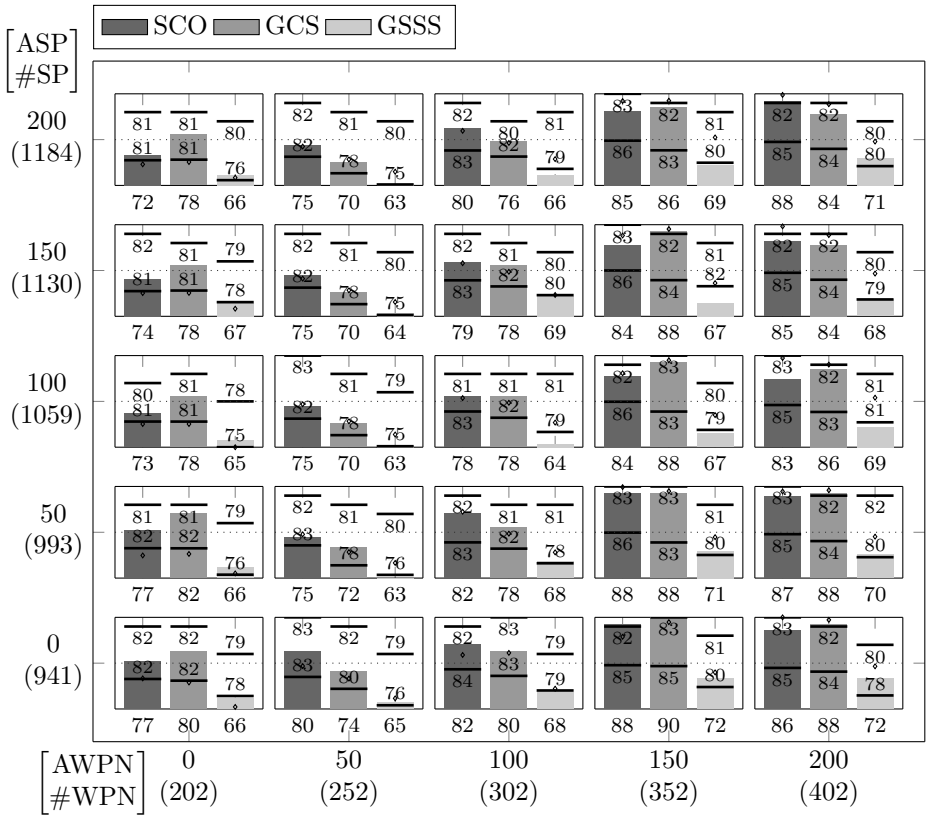


Figure 45: The #SP, quality, covered area and sum quality for the OF using different levels of SP-WPN-discretization.

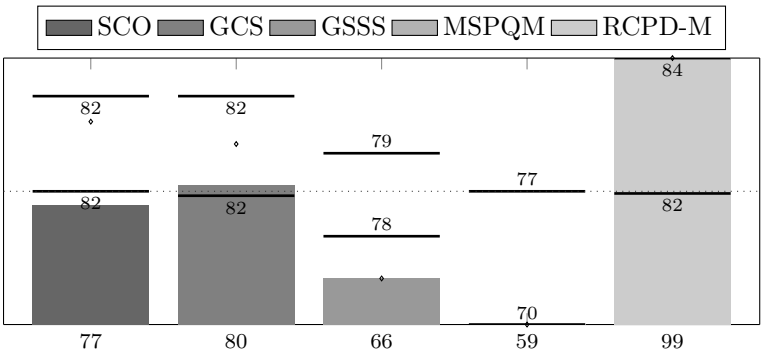


Figure 46: The #SP, quality, covered area and sum quality for the OF using no ASP and AWPN. Since the

INDEX