INFORMATION-DRIVEN SENSOR PATH PLANNING AND THE TREASURE HUNT PROBLEM

by

Chenghui Cai

Department of Mechanical Engineering and Materials Science
Duke University

Date: _____

Approved:

_____
Silvia Ferrari, Ph.D., Supervisor

_____
Lawrence Carin, Ph.D.

_____
Krishnendu Chakrabarty, Ph.D.

_____
Devendra Garg, Ph.D.

_____
Jeffrey Scruggs, Ph.D.

Dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Department of Mechanical Engineering and Materials Science
in the Graduate School of
Duke University

2008

ABSTRACT

(Engineering—Mechanical)

INFORMATION-DRIVEN SENSOR PATH PLANNING AND THE TREASURE HUNT PROBLEM

by

Chenghui Cai

Department of Mechanical Engineering and Materials Science
Duke University

Date: _____

Approved:

_____
Silvia Ferrari, Ph.D., Supervisor

_____
Lawrence Carin, Ph.D.

_____
Krishnendu Chakrabarty, Ph.D.

_____
Devendra Garg, Ph.D.

_____
Jeffrey Scruggs, Ph.D.

An abstract of a dissertation submitted in partial
fulfillment of the requirements for the degree
of Doctor of Philosophy in the Department of
Mechanical Engineering and Materials Science in the Graduate School of
Duke University

2008

Abstract

This dissertation presents a basic information-driven sensor management problem, referred to as treasure hunt, that is relevant to mobile-sensor applications such as mine hunting, monitoring, and surveillance. The objective is to classify/infer one or multiple fixed targets or treasures located in an obstacle-populated workspace by planning the path and a sequence of measurements of a robotic sensor installed on a mobile platform associated with the treasures distributed in the sensor workspace. The workspace is represented by a connectivity graph, where each node represents a possible sensor deployment, and the arcs represent possible sensor movements. A methodology is developed for planning the sensing strategy of a robotic sensor deployed. The sensing strategy includes the robotic sensor's path, because it determines which targets are measurable given a bounded field of view. Existing path planning techniques are not directly applicable to robots whose primary objective is to gather sensor measurements. Thus, in this dissertation, a novel approximate cell-decomposition approach is developed in which obstacles, targets, the sensor's platform and field of view are represented as closed and bounded subsets of an Euclidean workspace. The approach constructs a connectivity graph with observation cells that is pruned and transformed into a decision tree, from which an optimal sensing strategy can be computed. It is shown that an additive incremental-entropy function can be used to efficiently compute the expected information value of the measurement sequence over time.

The methodology is applied to a robotic landmine classification problem and the board game of CLUE®. In the landmine detection application, the optimal strategy of a robotic ground-penetrating radar is computed based on prior remote measurements and environmental information. Extensive numerical experiments show that this methodology outperforms shortest-path, complete-coverage, random, and grid search strategies, and is applicable to non-overpass capable platforms that must avoid targets as well as obstacles. The board game of CLUE® is shown to be an

excellent benchmark example of treasure hunt problem. The test results show that a player implementing the strategies developed in this dissertation outperforms players implementing Bayesian networks only, Q-learning, or constraint satisfaction, as well as human players.

Contents

List of Tables

List of Figures

Nomenclature

Symbols

$a(t_k)$	:	Action decision at time t_k
$\mathcal{A}$	:	Geometry of robotic platform
$\mathcal{B}_j$	:	j^{th} target, $j = 1, 2, \ldots, n$
$\mathcal{C}_{free}$	:	Robot free-configuration space
C_{ji}^{ℓ}	:	j^{th} ℓ category (suspect, weapon or room) cards dealt to i^{th} player
$\mathcal{CB}_j$	:	j^{th} C-obstacle, the Minkowski sum of $\mathcal{A}$ and $\mathcal{B}_j$, $j = 1, 2, \ldots, n$
$\mathcal{CT}_i$	:	i^{th} C-target, the Minkowski sum of $\mathcal{S}$ and $\mathcal{T}_i$, $i = 1, 2, \ldots, r$
d_{ij}	:	Distance between cell κ_i and κ_j
DT	:	Decision tree generated from T_r
E^0	:	Environmental conditions known *A-priori*
e_i	:	Evidence to infer y_i associated with $\mathcal{T}_i$
E_i	:	Environmental conditions associated with $\mathcal{T}_i$
$\mathcal{E}^0$	:	*A-priori evidence set* $\{v^0, E^0, M^0\}$
$\mathcal{F}_{\mathcal{A}}$	:	Moving Cartesian frame embedded in $\mathcal{A}$
F_i	:	Features of $\mathcal{T}_i$
$\mathcal{F}_{\mathcal{W}}$	:	Fixed Cartesian frame
$\mathcal{G}$	:	*Connectivity graph*
$J_y(\mathcal{T}_i \mid e)$	:	Error metric accounting for CL in classifying $\mathcal{T}_i$
κ_0	:	Robot initial cell, $q_0 \in \kappa_0$

κ_f	:	Robot final cell, $q_f \in \kappa_f$
$\mathcal{K}$	:	Finite set of discrete cells decomposed from $\mathcal{C}_{free}$
M^0	:	Set of prior IR sensor measurements
$\mathcal{M}_i$	:	Set of *test* variables $\{m_{i1}, \ldots, m_{iM}\}$ to infer y_i associated with $\mathcal{T}_i$
M	:	Set of posterior GPR measurements
q	:	Robot configuration
q_0	:	Robot initial configuration
q_f	:	Robot initial configuration
$\mathcal{S}$	:	Robotic sensor field of view
t_k	:	discrete time epoch indexed by k
$\mathcal{T}_i$	:	i^{th} target, $i = 1, 2, \ldots, r$
T_r	:	Connectivity tree obtained from $\mathcal{G}$
$u(t_k)$	:	Test decision at time t_k
$V(t_f)$	:	Expected observation profit or utility of σ^*
v^0	:	Prior IR sensor measurement mode
V_i	:	Sensor mode used to take measurements from $\mathcal{T}_i$
$\mathcal{W}$	:	Euclidean space
w_B	:	Weight to observation benefit B
w_D	:	Weight to cost of the sensor movement D
w_J	:	Weights to observation cost J
y_i	:	i^{th} *hypothesis* variable associated with $\mathcal{T}_i$
$\mathcal{Y}$	:	Mutually exclusive states of y_i, $\{y_i^1, y_i^2, \ldots, y_i^p\}$
y^r	:	Hypothesis variable denoting hidden room card
y^s	:	Hypothesis variable denoting hidden suspect room card
y^w	:	Hypothesis variable denoting hidden weapon card

σ	:	Sequence of decisions, inducing path corresponding to τ
σ^*	:	Sequence of optimal decisions, $\{u(t_k), a(t_k) \mid k = 0, \ldots, f\}$
τ_{cover}	:	Complete coverage path
τ_{rand}	:	Random coverage path
τ_{grid}	:	Path generated by fixed grid method
τ	:	Robot path induced by σ
τ^*	:	Robot optimal path or channel
$\eta_y(\sigma)$	:	Classification improvement per unit distance of policy σ
$\eta_{\mathcal{T}}(\sigma)$	:	Number of observed targets per unit distance of policy σ
ΔH_σ	:	Total entropy reduction given policy σ
ΔJ_y^σ	:	Total error reduction over policy σ
$\Omega(y^\ell)$	:	Domain of hidden hypothesis variable $y^\ell, \ell = s, w, r$
$(\cdot)^0$	:	Random variable(s) known *a priori*

Acronyms

BN	:	Bayesian network
CL	:	Confidence level
CSP	:	Constrained satisfaction player of CLUE® game
DS	:	Directed Search
EDG	:	Expected discrimination gain
EDGBS	:	Expected Discrimination Gain Based Search
EER	:	Expected entropy reduction
EERBS	:	Expected Entropy Reduction Based Search
EFIG	:	Expected fisher information gain
EFIGBS	:	Expected Fisher Information Gain Based Search

EHAD	:	Expected Hellinger affinity distance
EHADBS	:	Expected Hellinger Affinity Distance Based Search
EIPG	:	Expected information potential gain
EIPGBS	:	Expected Information Potential Gain Based Search
EQER	:	Expected quadratic entropy reduction
EQERBS	:	Expected Quadratic Entropy Reduction Based Search
FOV	:	Field of view
GPR	:	Ground penetrating radar
ICP	:	Intelligent computer player of CLUE® game
ID	:	Influence diagram
IR	:	Infrared
KL	:	Kullback-Leibler
PMF	:	Probability mass function
POMDP	:	Partially observable Markov decision process
RCP	:	Random computer player of CLUE® game
ROI	:	Region of interest
THP	:	Treasure Hunt Problem

Acknowledgements

I would first like to thank my advisor, Dr. Silvia Ferrari. Your help, financial support, and dedication to me over the course of my graduate studies were invaluable. Your consistent motivation kept me focused every day; your academic adventure enhances my interests and passion in research; and your kindness warms my heart when I feel alone for being away from my home country.

I would also like to give special mention those faculty who have served on my Ph.D. qualification, preliminary exam, and final defense committees for their time and effort: Dr. Devendra Garg, Dr. Michael Lavine, Dr. Ron Parr, Dr. Lawrence Carin, Dr. Krishnendu Chakrabarty, and Dr. Jeffrey Scruggs. In addition, I would like to thank those who have contributed to my better understanding of this research: Dr. Rafael Fierro, Dr. Tom Wettergren and Dr. Warren Fox. A special thank you to my great labmates and fellow graduate students, who were always available and willing to help along the way: Kelli A. Crews Baumgartner, Ming Qian, Gianluca Di Muro, Guoxian Zhang, and Casey Rubin.

I wish to dedicate this dissertation to my family. To my wife, Lixia. Your support, understanding, and love made this happen. I want to hold your hands in the rest of my life. To my little girl Ruoxin. You are so cute and lovely. Our talks over phone always fast renewed my energy when I was almost exhausted. I cannot wait to show you more about the world which is right in front of you.

Chapter 1

Introduction

This dissertation addresses the coupled problems of motion and measurement planning for a robotic sensor, so called treasure hunt problems. It is assumed that the robotic sensor referring to a sensor installed on a mobile robot platform navigates a workspace in order to make measurements from multiple targets or treasures whose features and classification must be inferred from the measurements. Sensor planning refers to the problem of determining a strategy for gathering sensor measurements to support a sensing objective, such as target classification. When the sensors are installed on robotic platforms an important part of the problem is planning the sensor path [1–4]. In fact, the robotic sensor path planning problem, which refers to plan the path and the measurements of a robotic sensor, is coupled with the robot motion planning, because the targets measured by the sensor depend on the path and motions of its platform. Several approaches have been proposed for planning the path of mobile robots with on-board sensors to enable navigation and obstacle avoidance in unstructured dynamic environments, e.g., [5–10]. However, these methods are not directly applicable to robotic sensors whose primary objective is to support a sensing objective, rather than to navigate a dynamic environment [11]. The reason is that these methods focus on how the sensor measurements pertaining the environment can best support the robot motion, rather than focusing on the robot motions that should be planned based on the measurement process and best support the sensing objective [11]. This dissertation addresses the problem of robotic sensor path planning in order to classify multiple targets distributed in an obstacle-populated workspace. The objective is to optimize overall sensing performance. This problem, known as the

treasure hunt [12], arises in many applications, such as robotic mine hunting [4, 13], cleaning [1], and monitoring of urban environments [14], manufacturing plants [15], and endangered species [16].

The most popular approaches to sensor path planning include coverage path-planning [2, 11], random [2], grid [17], and optimal search strategies [17, 18]. Optimal search strategies typically outperform other approaches in applications where *a-priori* information is available, such as sensor models, environmental conditions, and prior measurements [17]. However, they do not yet provide a systematic and general approach for sensor path planning in geometric sensing problems. Geometric sensing problems require a description of the geometry and position of the targets and of the sensor's field of view (FOV) [19]. Viewpoint planning has been shown by several authors to be an effective approach for optimally placing or moving vision sensors based on the target geometry and sensor FOV, using weighted functions or tessellated space approaches [19–21]. Probabilistic deployment has been shown to be an effective approach for searching for targets in a region of interest (ROI) by computing a search path based on the probability of finding a target in every unit bin of a discretized, obstacle-free workspace [1, 22, 23].

In this dissertation, an approximate cell decomposition approach is developed for solving the aforementioned treasure-hunt problem. Its advantage over existing sensor path planning techniques is that it takes into account the motion and geometry of closed and bounded subsets of an Eucledian space representing the sensor's platform and field of view, as well as the geometry and position of multiple fixed targets and obstacles in the ROI. Traditionally, approximate cell decomposition has been used to plan the motions of a robot with geometry $\mathcal{A}$, in order to avoid collisions with multiple fixed obstacles in a workspace $\mathcal{W}$ [24]. In this dissertation, the approach in [24] is modified to plan the motions of a robotic sensor with FOV $\mathcal{S}$ and plat-

form $\mathcal{A}$, in order to make measurements from multiple targets in $\mathcal{W}$ (comprising the ROI), while avoiding collisions with the obstacles in $\mathcal{W}$. Since the sensor is installed on-board the robot, the configuration of both $\mathcal{A}$ and $\mathcal{S}$ can be specified with respect to the same coordinate frame, embedded in $\mathcal{W}$. Then, the free-configuration space is decomposed to obtain a connectivity graph with observation cells that each enable a unique set of measurements from one or more targets in $\mathcal{W}$. This novel approximate-and-decompose procedure can be considered as a systematic approach for constructing so-called *detection cells*, used for information-driven sensor planning in [25, 26].

Information-driven sensor planning refers to sensor planning based on the expected information value or *benefit* of sensor measurements, and has been shown by several authors to be a general and effective framework for computing the expected measurements' value in sensor planning problems [25–28]. While robot path planning typically aims to optimize a deterministic additive function such as Eucledian distance, sensor path planning aims to optimize a stochastic sensing objective that is not necessarily additive. Also, the sensor's position and parameters (or mode) must be planned prior to obtaining sensor measurements. Therefore, while the measurements ultimately determine performance with respect to the sensing objective (e.g., classification), they cannot be factored into the planning problem [25–29]. Recently, the authors showed that using an additive expected entropy reduction (EER) function instead of relative entropy [25, 26] leads to improved target classification in non-Gaussian sensor fusion [30]. In this dissertation, EER, a type of incremental entropy, is used to formulate the expected value of the sensor measurements in terms of a posterior probability mass function (PMF) obtained from *a-priori* information (Section 3). Further numerical comparison of different information measure of the expected value of the posterior sensor measurements is implemented in Section 3.2.

A procedure is presented for pruning and transforming the connectivity graph into a decision tree that is used to determine the sensing strategy with maximum expected measurement profit.

In summary, the advantages of three approaches mentioned above, robot path planning, geometric sensing and information-driven sensor planning, are combined in this dissertation to solve the proposed treasure hunt problem, in which the path of a robotic sensor is planned based not only on the geometry of its platform, but also that of its FOV, so that one can optimize its measurement sequence and motions, by taking into account the intersections of its FOV with the targets as well as the expected value of information associated with the measurements. The treasure hunt problem and the robotic sensor planning methodology are demonstrated through the board game of CLUE® and a demining application, where CLUE® is a singleton hypothesis variable treasure hunt problem and the demining application contains multiple hypothesis variables. A computerized game of CLUE® is developed and represents an excellent benchmark for the treasure hunt problem. Our results show that the methodology developed outperforms both human players and a computer player implementing Bayesian networks only [31], Q-learning [12] and constraint satisfaction approach [32]. In the demining application, it is shown that the proposed method accounts not only for the geometry of the obstacles and the robots, but also for the geometry of the targets and of the sensor field of view. This method allows to obtain global optimal solution for planning both the robot motions and the sensor measurements simultaneously. The proposed method also allows to account for minefield environmental conditions and prior IR sensor information in planning the optimal sensor strategy, and achieves much better overall efficiency than other methods, such as A*, fixed grid, complete and random coverage.

Chapter 2

Problem Formulation and Assumptions

The purpose of many surveillance systems deploying sensors mounted on robotic platforms is to infer one or more hidden variables from the measurements and fusion of multiple target features. A hidden or *hypothesis* variable, y_i, may represent a target's class or typology, and the measurements may represent physical properties that are observable provided the target lies within the sensor's field of view. The outcomes of the measurements are unknown *a priori*, and only by visiting the target's site they can be obtained.

Let $\mathcal{W}$ denote a Euclidean space that is populated with r fixed targets $\mathcal{T}_i$, $i = 1, 2, \ldots, r$, and n fixed obstacles $\mathcal{B}_j$, $j = 1, 2, \ldots, n$, such that $\mathcal{T}_i \cap \mathcal{B}_i = \varnothing$ for all i, j. Assume that to each target $\mathcal{T}_i$ there is associated one hidden variable y_i that is discrete and, possibly, random, with a finite set of mutually exclusive states denoted by $\mathcal{Y} = \{y_i^1, y_i^2, \ldots, y_i^p\}$. Although it cannot be directly measured or observed, y_i can be inferred from a set of *test* variables, $\mathcal{M}_i = \{m_{i1}, \ldots, m_{iM}\}$, through a known joint probability distribution: $P(y_i, \mathcal{M}_i)$. Every variable $m_{i\ell}$ also is random and discrete and has N_ℓ possible outcomes, where $m_{i\ell}^k$ denotes the k^{th} outcome of measurement $m_{i\ell}$.

A map of all potential targets' and obstacles' geometry and location is provided *a priori*. The robotic platform is denoted by $\mathcal{A}$, and its configuration, q, specifies the position and orientation of a moving Cartesian frame $\mathcal{F}_\mathcal{A}$, embedded in $\mathcal{A}$, with respect to a fixed Cartesian frame, $\mathcal{F}_\mathcal{W}$. The sensor $\mathcal{S}$ is mounted on $\mathcal{A}$ with a fixed position and orientation that can be specified through the same moving frame of reference $\mathcal{F}_\mathcal{A}$. Where, $\mathcal{S}$ and $\mathcal{A}$ are closed and bounded subsets of $\mathcal{W}$. In this

dissertation, we address the problem of planning the path of $\mathcal{A}$ for the purpose of enabling sensor measurements from the targets, while avoiding collisions with the obstacles.

The robot free-configuration space $\mathcal{C}_{free}$ is decomposed into a finite set of discrete cells, $\mathcal{K} = \{\kappa_1, \kappa_2, \ldots\}$. An *observation cell* in $\mathcal{K}$ is a convex polygon in $\mathcal{C}_{free}$ with the property that every configuration in it enables the observation of at least one target in $\mathcal{W}$. A *void cell* in $\mathcal{K}$ is a convex polygon in $\mathcal{C}_{free}$ that does not enable any target measurements. A methodology is presented in Section 4.1 for obtaining the aforementioned decomposition. The robot may visit only one cell at a time and, after visiting a cell κ_i, the sensor can move to an adjacent cell, κ_j, by incurring a cost $d_{ij} = d_{ji}$. The adjacency relationships between these cells are provided by a *connectivity graph*, $\mathcal{G}$ (as shown in Section 4.1). Due to energy and time considerations, only a subset of targets in $\mathcal{W}$ may be visited by the sensor. Since the measurements outcomes are unknown *a priori*, the expected profit of the measurements obtained from $P(y_i, \mathcal{M}_i)$ is used to plan the robotic sensor actions using *hypothesis-driven* decision making [33], also known as *pre-posterior decision* analysis [34] .

For convenience, a discrete time t_k is used as an index for the sequential nature or *causality* of the sensor movements from one cell to the next. Suppose the sensor is inside cell κ_i at time t_k. Then, the cells that can be visited subsequent to κ_i, at a time t_{k+1}, are all the cells that are adjacent to κ_i in G. At every time t_k, the sensor makes a decision, $u(t_k)$, on whether to make an available measurement, and a decision, $a(t_k)$, on which cell to move to at time t_{k+1}. The robotic sensor performance is defined as the profit of the observation performed at t_k:

$$R(t_k) = w_B \cdot B(t_k) - w_J \cdot J(t_k) - w_D \cdot D(t_k) \tag{2.1}$$

Where, B is the observation benefit, J is the observation cost, D is the cost of the sensor movement: $D(t_{k+1}) = D(\kappa_i, \kappa_j) = d_{ij}$, and w_B, w_J and w_D are the weights.

As shown in Section 3.3, B can be formulated in terms of an information reward function. Therefore, an optimal policy for the robotic sensor can be obtained by solving the following problem:

Problem 2.0.1 (Treasure Hunt Problem) *Given a layout $\mathcal{W}$ and a joint probability distribution, $P(y_i, \mathcal{M}_i)$, for any $\mathcal{T}_i \subset \mathcal{W}$, find the sequence of decisions $\sigma^* = \{u(t_k), a(t_k) \mid k = 0, \ldots, f\}$ that maximizes the expected observation profit,*

$$V(t_f) = E\left\{\sum_{k=0}^{f} R(t_k)\right\}, \tag{2.2}$$

along an obstacle-free channel $\tau^* \equiv \{\kappa_0, \kappa_i, \ldots, \kappa_j, \kappa_f\}$, for a robotic sensor $\mathcal{S}$ installed on a platform $\mathcal{A}$, that must travel from an initial configuration $q_0 \in \kappa_0$ to a final configuration $q_f \in \kappa_f$.

Chapter 3

Information-Driven Sensor Planning

Information-driven sensor planning aims at making decisions regarding the optimal sensor type, mode, or configuration by formulating the sensing objectives through information theory. An underlying difficulty in sensor planning consists of assessing the value of sensor measurements prior to observing their outcomes. Several authors have proposed using information-theoretic functions for sensor planning. Schmaedeke used relative information content, which actually was relative entropy, to solve a multisensor-multitarget assignment problem [28]. Kastella used relative entropy to manage agile sensors for target detection and classification with underlying Gaussian probability distributions [25, 26]. Zhao investigated information objective functions such as entropy and Mahalanobis distance measure for sensor collaboration applications [27]. Recently, it was shown in [30] that using incremental entropy instead of relative entropy leads to improved target classification and feature inference, when the underlying distributions are non-Gaussian and the measurements accuracy varies significantly among targets. Therefore, in this dissertation, incremental entropy is used to formulate the observation benefit function B, as shown in Section 3.3.

A common approach to implementing information-theoretic functions for sensor planning is to confine each target to a discrete cell [25]. Then, it can be assumed that when the sensor is directed toward a single cell (indexed by j) it produces a set of discrete or continuous measurements that depend on the target found in the cell, and here are denoted by $\mathcal{M}_j$. Although the target characteristics are unknown *a priori*, if the posterior probabilities $P(\mathcal{M}_j \mid y_j)$ and the priors $P(y_j)$ are given, the problem of planning the sensor mode and the cell to be measured can be solved by determin-

ing the cell with maximum relative-entropy [25]. These existing methods utilize an abstract notion of target cells and do not provide any guidelines for systematically defining and computing the cells in terms of physical parameters. Therefore, they cannot be readily applied to plan the sensor movements relative to the targets, or to devise a strategy for pointing the sensor toward the target of interest. Also, since they implement a non-additive relative-entropy objective function, they can select only one optimal cell (or target) at a time from an available set.

In this research, we develop a methodology that defines cells as geometric subsets of a robot configuration space and, systematically, computes a discrete-cell representation of a given workspace $\mathcal{W}$, based on the robotic sensor geometries $\mathcal{A}$ and $\mathcal{S}$, respectively (Section 4.1). Also, since the sensor must visit an optimal sequence of targets, we implement the incremental entropy approach that we developed in [30], and that is reviewed in Section 3.3.

3.1 Review of Bayesian Network Sensor Modeling

A probabilistic model of the sensor measurements is obtained in the form of a Bayesian network (BN), using the approach in [35]. BNs map causal-effect relationships among all relevant variables by learning the underlying joint probability distributions from data and, possibly, heuristic arguments. They can be used for modeling a generic sensor measurement process by selecting the BN nodes to represent variables that influence the measurements outcomes, and by learning the BN arcs and parameters from prior measurement data. The BN nodes are selected by considering the following sets of variables: the operating parameters or mode V, the environmental conditions E, the measurements $\mathcal{M}$, and the actual target features F that must be inferred from $\mathcal{M}$. Also, when the sensor measurements are used for classification, the target category is represented by a variable or node y. In this

dissertation, upper case letters are used to define sets, and lower case letters are used to define variables. After the BN nodes $\mathcal{X}_S = \{V, E, \mathcal{M}, F, y\}$ have been selected, all of their possible instantiations must be identified such that they are countable and mutually exclusive.

The BN arcs and conditional probability tables (CPTs) are determined by a batch learning algorithm. In this approach [35], the initial architecture is specified based on expert knowledge of the sensor working principles, as shown in Fig. 3.1. Subsequently, the final BN arcs and CPTs that best capture the measurement process are obtained from a database of prior sensor measurements. This database consists of several training cases in which all variables in $\mathcal{X}_S$ are instantiated, and is constructed by obtaining sensor measurements from several known targets, under known environmental conditions [36]. After training is completed, the BN model specifies the joint probability distribution underlying the sensor measurements in terms of the following factorization,

$$P(\mathcal{X}_S) = P(V, E, \mathcal{M}, F, y) = \prod_{x_l \in \mathcal{X}_S} P(x_l \mid pa(x_l)) \tag{3.1}$$

where $pa(x_i)$ denotes the parents of a node x_i in $\mathcal{X}_S$.

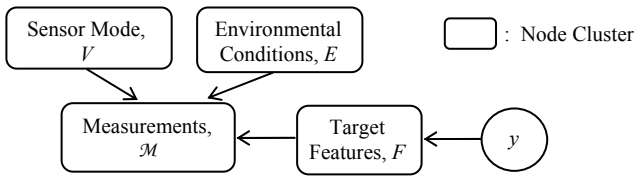


Figure 3.1: Initial architecture of BN sensor model

When measurements are obtained from an unknown target, $\mathcal{T}_i$, the outcomes of $\mathcal{M}_i$ are known and, together with any information pertaining the mode, V_i, and environmental conditions, E_i, they provide the evidence e_i for the BN model of the sensor used to obtain the measurements. Thus, the BN model can be used to infer

the features, F_i, and classification, y_i of $\mathcal{T}_i$, by computing $P(F_i, y_i \mid e_i)$ [35]. In many applications, such as demining, the measurements obtained from multiple and heterogeneous sensors must be obtained and fused in order to achieve satisfactory classification performance. In this case, the BN models of each sensor type are used in combination with by the Dempster-Shafer (DS) rule of evidence combination [37, 38] to obtain a fused posterior probability distribution for F_i and y_i, as shown in [35].

In summary, the BN model of a sensor provides a convenient representation for the probability distributions underlying the measurement process, based on prior data and expert knowledge. The BNs can be used to infer target features and classifications from known measurements. Also, they can be used to compute the observation benefit function, B, for the sensor measurements before the actual measurements become available, i.e. *a posteriori*, as shown in the next section.

3.2 Information Measure Comparisons

In this section, different information measures are compared so as to select an effective one which will yield good classification performance for path and sensor planning in THPs. A robotic demining problem is applied to implement the numerical comparisons. In this application, first an IR (infrared) sensor mounted on an airplane flying over the minefield was used to obtain prior information, and then autonomous ground vehicles (AGVs) carrying a posterior Ground Penetrating Radar (GPR) sensor moved around to improve the discovery and classification of objects buried underground. Both prior and expected posterior sensor information was used as feedback to the vehicle for control and planning purposes. The purpose of GPR sensor measurements is to reduce the uncertainty in the hypothesis variable y_i and improve its classification. Let M denote a new (posterior) set of GPR measurements, given an *a-priori evidence set* $\mathcal{E}^0 = \{v^0, E^0, M^0\}$, which may include known environmental

conditions, as well as the measurements M^0 and mode v^0 of a previously-deployed IR sensor. The superscript $(\cdot)^0$ denotes one or more random variables whose values are known *a priori*.

The problem of sensor planning is to determine the best way to task a sensor or group of sensors when each sensor may have many modes and search patterns. Typically, the sensors are used to gain information about the kinematic state (e.g. position and velocity) and identification of a group of targets. Recently, Cramér-Rao bounds have been used to control the measurement sequence in a sensor management setting [39, 40]. The bound is also known as the Cramér-Rao inequality or the information inequality [41] and has close relation to Fisher information measure which has been used for optimizing a sampling design [42]. In its simplest form, the bound states that the variance of any unbiased estimator is at least as high as the inverse of the Fisher information measure.

Other information measures as a mean of information-driven sensor planning has been proposed by several authors for computing the expected measurements' value [25–28]. In the sensor planning problems using Bayesian estimation, reduction in entropy of the posterior distribution that is expected to be induced by the measurement is a good measure of the quality of a sensing action. Thus, information theoretic sensor planning methodologies strive to take the sensing path or decision that maximizes the expected information gain. The possible sensing decisions are enumerated, the expected information gain for each measurement is calculated, and the decision that yields the maximal expected gain is chosen. Several different information measures, such as simple heuristics, entropy and discrimination (relative entropy or Kullback-Leibler divergence), are compared in an application of dynamic sensor collaboration in ad hoc sensor networks [27]. Other related applications of discrimination gain based on a measure of relative entropy, the Kullback-Leibler (KL)

divergence, are described in [25,26,28]. Especially, in [26], a quite general information measure called the Rényi information divergence [43], also known as the α-divergence, is utilized to guide scheduling sensors for multiple target tracking applications. In the limiting case of $\alpha \mapsto 1$, the Rényi divergence becomes the commonly utilized (KL) discrimination. A partially observable Markov decision process (POMDP) was proposed for sensing a hidden target from sensors on a platform by minimizing the expected cost, and the classification actions were taken by weighing the expected cost of performing future sensing actions with expected future reduction in the Bayes risk [29].

Information measures such as mutual information and entropy reduction are also commonly used as learning metrics that have been used in the machine learning literature. Quadratic entropy and information potential are used as metrics in unsupervised learning [44]. Maximizing mutual information is applied in unsupervised neural networks learning [45].

In this section, different information measures are defined below and then their numerical comparisons are implemented via the demining application described in Chapter 6. While the details of this demining application will be explained in Chapter 6, the purpose of considering this demining problem in this section is to compare the performance of different information measures in sensor planning applications.

Information entropy is a function that represents the uncertainty or lack of information in a discrete and random variable that can be defined with respect to a variable's probability distribution. The *entropy* $H(x)$ of a discrete random variable x is defined by [41]:

$$H(x) = -\sum_{x} P(x) \log_2 P(x). \tag{3.2}$$

The *conditional entropy* $H(y|x)$ where y is the *hypothesis* variable is defined as [41]:

$$H(y|x) = -\sum_{x} P(x)H(y|x) = -\sum_{x} P(x)\sum_{y} P(y|x)\log_2 P(y|x). \tag{3.3}$$

Consider instead the *incremental entropy* or conditional mutual information [41] that for three discrete and random variables y, x_1, and x_2 is defined as:

$$\begin{aligned} I(y; x_2|x_1) &= H(y|x_1) - H(y|x_1, x_2) && (3.4)\\ &= E_{y,x_1,x_2}\left\{\log_2 \frac{P(y, x_2|x_1)}{P(y|x_1)P(x_2|x_1)}\right\} && (3.5) \end{aligned}$$

where, E denotes the expectation with respect to its subscript. Since the conditional entropy $H(y \mid \mathcal{E}^0, M)$ cannot be determined prior to measuring M, the EER,

$$\Delta H(y; M|\mathcal{E}^0) \equiv H(y|\mathcal{E}^0) - \sum_{M}\left[H(y|\mathcal{E}^0,\ M)P(M|\mathcal{E}^0)\right] = H(y|\mathcal{E}^0) - E[H|\mathcal{E}^0, M]. \tag{3.6}$$

The proposed EER actually equals incremental entropy or conditional mutual information $I(y; M|\mathcal{E}^0) = H(y|\mathcal{E}^0) - H(y|\mathcal{E}^0,\ M)$, since $E[H|\mathcal{E}^0, M] = H(y|\mathcal{E}^0,\ M)$.

The calculation of information gain between two densities f_1 and f_0 is done using the Rényi information divergence:

$$D_\alpha(f_1\|f_0) \ = \ \frac{1}{\alpha - 1}\ \log_2 \int f_1(x)\ f_0^{1-\alpha}(x)\ \ dx \tag{3.7}$$

The determining factor α is viewed as the degree of differentiation between the two densities under consideration. when $\alpha \mapsto 1$, the Rényi divergence becomes KL discrimination or relative entropy. The discrimination $D\,(y|\mathcal{E}^0)$ is defined as:

$$D\,(y|\mathcal{E}^0) = \sum_{y} P(y|\mathcal{E}^0)\log_2 \frac{P(y|\mathcal{E}^0)}{P(y)}. \tag{3.8}$$

It was empirically determined that if the two densities are very similar, i.e., difficult to discriminate, then the indexing performance of the Hellinger affinity distance ($\alpha = 0.5$) was observed to be better than the discrimination, or Kullback-Leibler (KL) divergence, or relative entropy. The expected discrimination gain (EDG) is defined as:

$$\Delta D(y; M|\mathcal{E}^0) \equiv \sum_M \left[D(y|\mathcal{E}^0,\ M)P(M|\mathcal{E}^0)\right] - D\left(y|\mathcal{E}^0\right) = E[D|\mathcal{E}^0, M] - D\left(y|\mathcal{E}^0\right). \tag{3.9}$$

The expected Hellinger affinity distance (EHAD) is defined as:

$$EHAD(y; M|\mathcal{E}^0) \equiv \sum_M \left[D_{0.5}(P(y|\mathcal{E}^0,\ M)\|P(y|\mathcal{E}^0))P(M|\mathcal{E}^0)\right]. \tag{3.10}$$

The Fisher information is defined as:

$$J(\theta) = E_\theta \left[\frac{\partial \ln f(x;\theta)}{\partial \theta}\right]^2. \tag{3.11}$$

By the Cramér-Rao Inequality, the mean squared error of any unbiased estimator $T(x)$ of the parameter θ is lower bounded by the reciprocal of the Fisher information, i.e.,

$$J(\theta) \geq \frac{1}{var(T)} = \frac{1}{E(x^2) - E(x)^2} \tag{3.12}$$

For the case, when a Gaussian distribution can approximate the posterior, the Fisher information satisfies:

$$J(\theta) = \frac{1}{var(T)}. \tag{3.13}$$

In the demining problem, the parameter to be estimated is the hypothesis variable y and the observable random variables are the sensor measurements. Since all the random variables in the demining problem are discrete, it is hard to compute the

Fisher information in 3.11. Therefore, the Fisher information $J(y|\mathcal{E}^0, M)$ can be approximated as:

$$J(y|\mathcal{E}^0, M) \simeq \frac{1}{var(y|\mathcal{E}^0, M)}. \tag{3.14}$$

The expected fisher information gain (EFIG) can be defined as:

$$\Delta J(y; M|\mathcal{E}^0) \equiv \sum_M \left[J(y|\mathcal{E}^0, M)\ P(M|\mathcal{E}^0)\right] - J(y|\mathcal{E}^0). \tag{3.15}$$

The Information Potential (IP), denoted by $V(y|\mathcal{E}^0,\ M)$, of $P(y|\mathcal{E}^0,\ M)$ is defined as:

$$V(y|\mathcal{E}^0,\ M) = \sum_{i=1}^{p} P(y = i|\mathcal{E}^0,\ M)^2, \tag{3.16}$$

where y is with a finite range $\mathcal{Y} = \{y^1, \ldots, y^p\}$. The expected information potential gain (EIPG) is defined as:

$$\Delta V(y; M|\mathcal{E}^0) \equiv \sum_M [V(y|\mathcal{E}^0, M)\ P(M|\mathcal{E}^0)] - V(y|\mathcal{E}^0). \tag{3.17}$$

The quadratic entropy of y given $\{\mathcal{E}^0, M\}$ can be defined as:

$$H_{R2}(y|\mathcal{E}^0, M) = -\log_2 V(y|\mathcal{E}^0,\ M). \tag{3.18}$$

The expected quadratic entropy reduction (EQER) is defined as:

$$\Delta H_{R2}(y; M|\mathcal{E}^0) \equiv H_{R2}(y|\mathcal{E}^0) - \sum_M [H_{R2}(y|\mathcal{E}^0, M)\ P(M|\mathcal{E}^0)]. \tag{3.19}$$

The terms needed for computing the information measures defined above can be obtained via IR and GPR sensor models in Fig. 6.1 and BN classifier in Fig. 6.2. For comparison, the 98 target cells in the example mine field shown in Fig. 7.8 are searched using the following three methodologies:

1. Directed Search (DS): advance through the cells in the same order for every frame, taking one measurement over each cell.

2. Expected Discrimination Gain Based Search (EDGBS): direct the sensor to search the cells with the highest expected discrimination gain, taking one measurement over each cell.

3. Expected Entropy Reduction Based Search (EERBS): direct the sensor to search the cells with the highest expected entropy reduction, taking one measurement over each cell.

4. Expected Hellinger Affinity Distance Based Search (EHADBS): direct the sensor to search the cells with the highest expected Hellinger affinity distance, taking one measurement over each cell.

5. Expected Fisher Information Gain Based Search (EFIGBS): direct the sensor to search the cells with the highest expected Fisher information gain, taking one measurement over each cell.

6. Expected Quadratic Entropy Reduction Based Search (EQERBS): direct the sensor to search the cells with the highest expected quadratic entropy reduction, taking one measurement over each cell.

7. Expected Information Potential Gain Based Search (EIPGBS): direct the sensor to search the cells with the highest expected information potential gain, taking one measurement over each cell.

Assume that the GPR sensor is only allowed to make a fixed number of measurements, due to energy and time limitations. The goal is to direct the GPR to search the target cell sequence that produces the maximum improvement of average IR-GPR

sensor fusion classification accuracy, using fixed GPR measurement times. If the classification is correct, then the classification accuracy is 1; otherwise 0. The results are shown in Fig. 3.2 and 3.3. In Fig. 3.2, the average classification accuracy which is defined as the sum of classification accuracy over the number of measured targets is shown on the ordinates. In Fig. 3.3, average classification accuracy gain which is defined as average classification accuracy after both IR and GPR measurements minus average classification accuracy only with IR measurements.

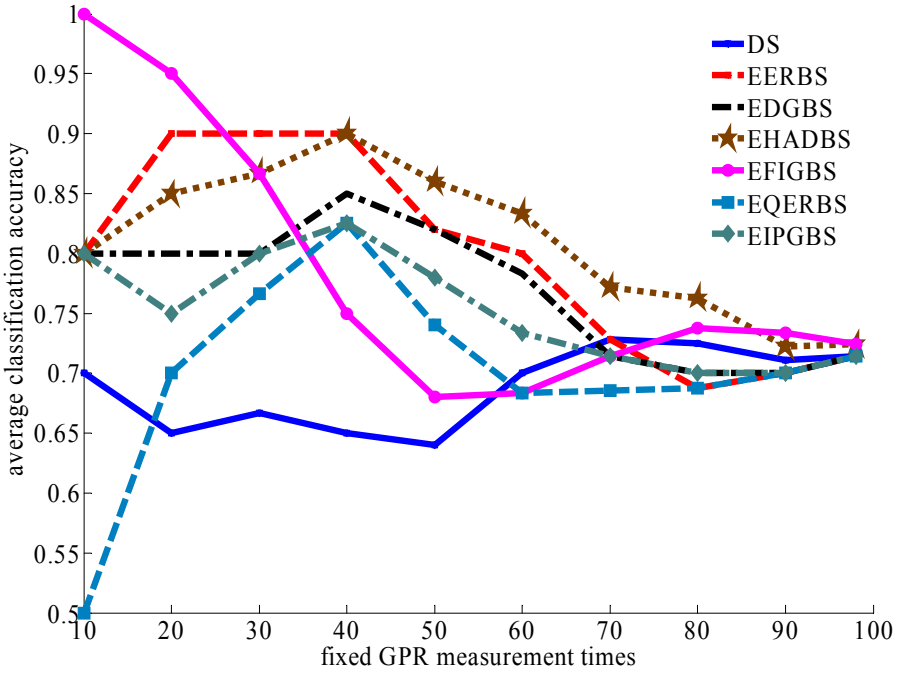


Figure 3.2: Average classification accuracy using seven searching technique.

As is seen from Fig. 3.2 and 3.3, considering both the classification accuracy and classification gain, Expected Hellinger Affinity Distance Based Search (EHADBS) and Expected Entropy Reduction Based Search (EERBS) are the best two. In Fig. 3.2, although the information measure of EFIG yields the highest average classification accuracy given that the amount of fixed GPR measurement times is less than 25, this information measure brings worse classification gain, shown in Fig. 3.3, than

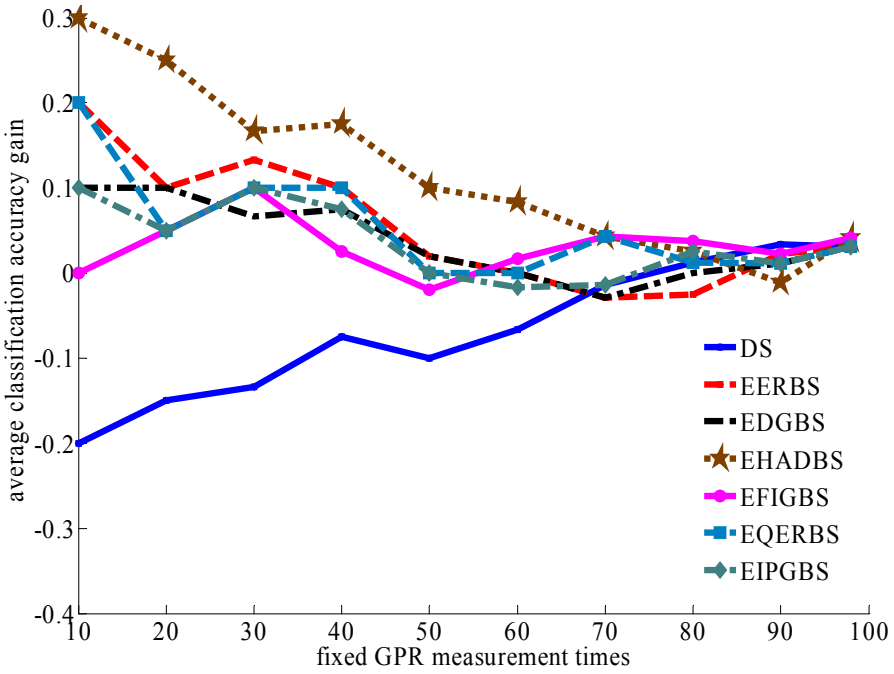


Figure 3.3: Average classification accuracy gain using seven searching technique.

many other measures, such as EDG, EER, EHAD, etc. Although in the demining problem, EHADBS is better than EERBS in achieving better classification gain, EER is still selected in this dissertation to formulate the expected value of the sensor measurements due to the following three reasons. The first reason, which is the most important, is that the EER is the mutual information in nature and is shown to be an additive function of a sequence of sensor measurements in Theorem 5.3.1 and [46]. This property brings several advantages. For instance, the principle of optimality is satisfied [47] and EER can be used to efficiently compute the expected information value of the measurement sequence over time. Secondly, accounting for a tradeoff between the classification accuracy and classification gain, the performance difference between EER and EHAD is not big. In Fig. 3.2, when the amount of fixed GPR measurement times is less than 40, EER yields better classification accuracy, whereas EHAD gives better classification gain than EER. Finally, it is easy to substitute EER

with EHAD while the sensor planning methodology proposed in this dissertation keeps the same.

On the other hand, EER may show its disadvantage in some classification applications where high classification confidence level is strongly favored. For example, in the robotic demining problem, if the misclassification of a true mine as a clutter is more concerned on than the misclassification of a clutter as a mine and the claim of a clutter should be based on high confidence level, EER may not be the best choice, because the change from a probability distribution of a relatively high confidence level to one of the favored higher confidence level will result in little reduction in entropy. In this case, a possible new information measure can be designed based on EER so that the new measure is piece-wise, e.g., having a large gain for a small increase in high confidence level. Another example is about the information measure of EHAD which describes the "informal" distance between two distributions. Assume that the prior sensor measurement gives a correct classification. A false posterior sensor measurement may bring an unfavored estimate and will result in a big gain in Hellinger affinity distance between the two posterior distributions obtained by both prior and posterior sensors, and only by prior sensor, but the correct classification may be lost after the posterior sensor measurement is taken. In brief, we have to admit that different information measures have different advantages and disadvantages. The choice or design of a suitable information measure really depends on the specific problem and its objectives. As this dissertation focuses on proposing general methodologies to solve the treasure hunt problem, it is easy to substitute one information measure with another while the systematic methodologies keep unchanged.

It can be shown in Appendix A that theoretically, $\Delta H(y; M|\mathcal{E}^0) = \Delta D(y; M|\mathcal{E}^0)$. However, we have two sensor models, IR sensor model and GPR sensor model. There are errors in these IR and GPR models which are learned from data. One of the ob-

vious evidences is that the prior probability of feature set T in IR model $P_{IR}(T)$ is not equal to the prior probability of feature set T in GPR model $P_{GPR}(T)$. In calculating EER and EDG in eq. 3.6 and 3.9, some terms have to be obtained from the two different models. Therefore, this results in that in the calculation, $\Delta H(y; M|\mathcal{E}^0) \neq \Delta D(y; M|\mathcal{E}^0)$. Their difference, $\Delta H(y; M|\mathcal{E}^0) - \Delta D(y; M|\mathcal{E}^0)$, reflects the model errors, and if viewed as a random variable, can be shown of mean 0.

3.3 Information Benefit Function

In an hypothesis-driven motion planning problem, the benefit of performing a sequence of measurements $Z_T = \{\mathcal{M}_1, \mathcal{M}_2, \ldots\}$ by moving the sensor in $\mathcal{W}$ during a period of time T is to decrease the uncertainty in the corresponding hypothesis variables $\{y_1, y_2, \ldots\}$. Since the measurements outcomes are unknown *a priori*, the benefit of observation must be defined over the posterior distributions $P(y_i \mid \mathcal{M}_i)$. Also, if the sensor measurements are fused with prior measurements, which may have been collected by a different type of sensor, the benefit of information must take those measurements into account as well.

As shown in [46], the incremental entropy is an additive function and the reduction in uncertainty brought about by a sequence of measurements Z_T is given by the sum

$$\begin{aligned} B(T) &= \sum_{\mathcal{M}_i \in Z_T} I(y_i; \mathcal{M}_i | \mathcal{M}_i^0) \\ &= \sum_{\mathcal{M}_i \in Z_T} \left\{ H(y_i|\mathcal{M}_i^0) - E_{\mathcal{M}_i}[H(y_i|\mathcal{M}_i) P(\mathcal{M}_i|\mathcal{M}_i^0)] \right\} \end{aligned} \tag{3.20}$$

Where, $\mathcal{M}_i^0$ denotes the set of prior sensor measurements available for target $\mathcal{T}_i$, and $Z^0 \equiv \{\mathcal{M}_1^0, \ldots, \mathcal{M}_r^0\}$ is the set of all prior measurements from $\mathcal{W}$. Finally, using the method presented in [30], all terms in the above benefit function can be formulated

in terms of CPTs available from the BN models of the sensors implemented. $B(T)$ is also called the expected entropy reduction (EER) brought about by a sequence of measurements Z_T.

Chapter 4

Methodology: Robotic Sensor Motion Planning

Existing robot motion planners have been devised to account for the presence of obstacles that the robot must avoid to reach a goal configuration in the workspace [48]. Cell decomposition is a well-known obstacle avoidance method that decomposes the free robot configuration space,

$$\mathcal{C}_{free} = \mathcal{C} \setminus \bigcup_{j=1}^{n} \mathcal{CB}_j = \{q \in \mathcal{C} \mid \mathcal{A}(q) \cap (\bigcup_{j=1}^{n} \mathcal{B}_j) = \varnothing\} \tag{4.1}$$

into a finite collection of non-overlapping convex polygons, or *cells*, within which a path free of obstacles can be easily generated. $\mathcal{CB}_j$ is a *C-obstacle* and $\bigcup_{j=1}^{n} \mathcal{CB}_j$ is the *C-obstacle region* [48]. The approximate rectangloid decomposition method, referred to as *approximate-and-decompose* [24], can be utilized to obtain an approximate cell decomposition of $\mathcal{C}_{free}$ for a convex polygonal robot $\mathcal{A}$ that is capable of translating and rotating in $\mathcal{W}$. In this method, cells of a predefined rectangloid shape are used to decompose the bounding and bounded approximations of the obstacles, until the connectivity of $\mathcal{C}_{free}$ is properly represented. Then, the union of the cells that are strictly outside the C-obstacle region are used to construct a non-directed connectivity graph representing the adjacency relationships between them. Finally, the connectivity graph is searched for the shortest path between an initial and a final configuration, q_0 and q_f.

4.1 Approximate-and-Decompose Method for Motion Planning in the Presence of Targets

In this section, a method based on the approximate-and-decompose approach [24] is developed for robotic sensor planning in the presence of targets. It is assumed that the sensor is mounted with a fixed position and orientation that can be specified through the same moving frame of reference as the robot, $\mathcal{F}_{\mathcal{A}}$, where:

Definition 4.1.1 (Field of View) *The field of view of a sensor mounted on $\mathcal{A}$ is a closed and bounded subset $\mathcal{S}(q) \subset \mathcal{W}$ such that every point $x \in \mathcal{S}(q)$ can be observed by the sensor when the robot occupies the configuration q specifying the position and orientation of a moving Cartesian frame $\mathcal{F}_{\mathcal{A}}$, embedded in $\mathcal{A}$, with respect to a fixed Cartesian frame, $\mathcal{F}_{\mathcal{W}}$.*

In order for a target to be observable by the sensor, it must intersect its field of view. Therefore, the subsets of $\mathcal{W}$ where the sensor can collect measurements can be defined similarly to *C-obstacles* [48], as follows:

Definition 4.1.2 (C-Target) *The target $\mathcal{T}_i$ in $\mathcal{W}$ maps in the robot's configuration space, $\mathcal{C}$, to the C-target region $\mathcal{CT}_i = \{q \in \mathcal{C} \mid \mathcal{S}(q) \cap \mathcal{T}_i \neq \varnothing\}$.*

Example 4.1.3 *Suppose the robot geometry $\mathcal{A}$ can be approximated by a rectangle, and the sensor field of view $\mathcal{S}$ is a triangle that has a fixed orientation θ_s with respect to $\mathcal{F}_{\mathcal{A}}$, as shown in Fig. 4.1.a. Let $\mathcal{A}$ be a robot that can translate freely but cannot rotate. Then, Fig. 4.1.b shows the geometry of the C-obstacle corresponding to an L-shaped obstacle, and Fig. 4.1.c shows the geometry of the C-target corresponding to a rectangular target.*

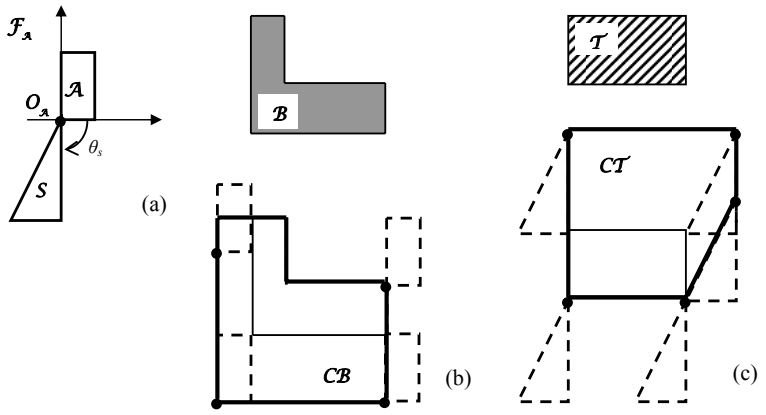


Figure 4.1: Example of C-obstacle (b) and C-target (c) obtained for a sensor with field of view $\mathcal{S}$ that is installed on a robot with geometry $\mathcal{A}$ at a fixed orientation θ_s (a).

While the robot geometry $\mathcal{A}$ must avoid collisions with the C-obstacle region, the sensor field of view $\mathcal{S}$ must intersect one or more C-targets in order for the sensor to make measurements. Since the sensor is installed on the robotic platform, the position and orientation of $\mathcal{S}$ depend on the position and orientation of $\mathcal{A}$. In other words, q specifies the position and orientation of both $\mathcal{S}$ and $\mathcal{A}$. Consequently, the targets measured by the sensor depend on the robot path, and the robot must avoid obstacles while searching for targets. Our approach consists of planning the sensor measurements in concert with the robot motion, and of treating the targets as the dual of the obstacles.

Although obstacles and targets may not intersect, the corresponding C-obstacles and C-targets intersect when a set of robot configurations that enables an observation, $\mathcal{S}(q) \cap \mathcal{T}_i \neq \varnothing$, causes the robot to collide with a nearby obstacle, $\mathcal{A}(q) \cap \mathcal{B}_j \neq \varnothing$. In this case, it follows that $\mathcal{CT}_i \cap \mathcal{CB}_j \neq \varnothing$ and, therefore, C-targets cannot be considered as observation cells. Additionally, although targets may not intersect, the corresponding C-targets may intersect when a set of robot configurations enables an observation of multiple targets. A simple example of workspace populated with C-

obstacles and C-targets is shown in Fig. 4.2. Therefore, a systematic approach is developed for dealing with both C-targets and C-obstacles, and obtain a decomposition of $\mathcal{C}_{free}$, $\mathcal{K}$, that contains and unequivocal subdivision of void and observation cells. A void cell is a convex polygon κ in $\mathcal{C}_{free}$ with the property that none of the targets are observable from any of the configurations in κ. Also, we introduce the formal definition:

Definition 4.1.4 (Observation Cell) *An observation cell is a convex polygon $\bar{\kappa}$ in $\mathcal{C}_{free}$ with the property that every configuration in $\bar{\kappa}$ enables the same non-empty set of observations $Z(\bar{\kappa}) = \{\mathcal{M}_i \mid q \in \bar{\kappa},\ q \in \mathcal{CT}_i\}$.*

We present the approach for $\mathcal{C} = \mathcal{W} \times [\theta,\ \theta']$, where $\mathcal{W} \subset \mathbb{R}^2$, and $[\theta,\ \theta']$ is the range of robot orientations. $\mathcal{C}$ is decomposed by first partitioning the interval $[\theta,\ \theta']$ into ν maximal closed subintervals $\mathcal{I}_u$, with $u = 1, \ldots, \nu$. Then, the bounding approximations of C-obstacles and the bounded approximations of C-targets in the rectangloid $\mathcal{W} \times \mathcal{I}_u$ are obtained via the outer projection of C-obstacle and the inner projection of C-targets into $\mathbb{R}^2$, respectively. The void rectangloid cells are extracted from the complement of the union of the bounding approximations of C-obstacles with the bounded approximations of C-targets in $\mathcal{W} \times \mathcal{I}_u$. The observation cells are rectangloids extracted from the bounded approximations of C-targets.

Let x, y, and θ denote the coordinates and orientation in $\mathcal{F}_{\mathcal{W}}$. Then, $\kappa = [x_\kappa, x'_\kappa] \times [y_\kappa, y'_\kappa] \times [\theta_\kappa, \theta'_\kappa]$ denotes a rectangloid cell in $\mathcal{C}$. We denote the intersections of κ with the C-obstacles and C-targets by $\mathcal{CB}_j[\kappa] = \mathcal{CB}_j \cap \kappa$ and $\mathcal{CT}_i[\kappa] = \mathcal{CT}_i \cap \kappa$, respectively. Then, the following approximation are obtained, as shown in Fig. 4.2.b:

Definition 4.1.5 (Bounding Rectangloid Approximation) *A bounding rectangloid approximation of $\mathcal{CB}_j[\kappa]$, denoted by $\mathcal{RB}_j[\kappa]$, is a collection of non-overlapping rectangloids $\mathcal{R}_v, v = 1, \ldots, p$, whose union contains $\mathcal{CB}_j[\kappa]$.*

Definition 4.1.6 (Bounded Rectangloid Approximation) *A bounded rectangloid approximation of $\mathcal{CT}_i[\kappa]$, denoted by $\mathcal{R}'\mathcal{T}_i[\kappa]$, is a collection of non-overlapping rectangloids $\mathcal{R}'_v, v = 1, \ldots, p'$, whose union is contained in $\mathcal{CT}_i[\kappa]$.*

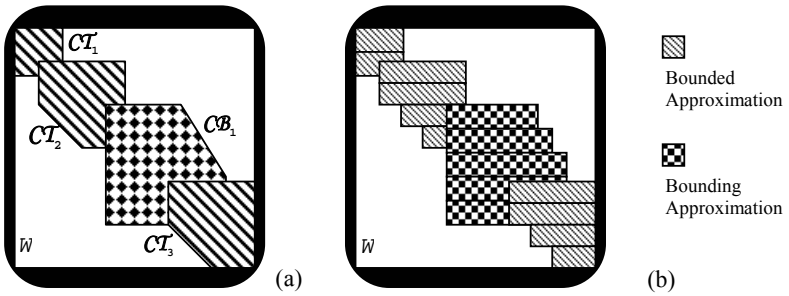


Figure 4.2: Simple example of workspace $\mathcal{W}$ populated with both C-obstacles and C-targets (a) and corresponding bounded and bounding approximations (b).

The above approximations are computed and decomposed for all obstacles and targets in $\mathcal{W}$ using the following steps:

1. Decompose $[\theta, \theta']$: The range of orientations is cut into non-overlapping intervals $\mathcal{I}_u = [\gamma_u, \gamma_{u+1}]$, with $u = 1, \ldots, \nu$, $\gamma_1 = -\pi/2$, $\gamma_{\nu+1} = \pi/2$, and $\nu \geq 1$. Then, let $\kappa^u = [x_\kappa, x'_\kappa] \times [y_\kappa, y'_\kappa] \times \mathcal{I}_u$.

2. Compute $\mathcal{RB}_j[\kappa^u]$ and $\mathcal{R}'\mathcal{T}_i[\kappa^u]$: For every $u = 1, \ldots, \nu$ and $j = 1, \ldots, n$, compute the outer projection,

$$\mathcal{OCB}_j[\kappa^u] = \{(x, y) \mid \exists \theta \in \mathcal{I}_u : (x, y, \theta) \in \mathcal{CB}_j[\kappa^u]\} \tag{4.2}$$

and generate bounding rectangloid approximation of $\mathcal{OCB}_j[\kappa^u] \times \kappa^u$. For every $u = 1, \ldots, \nu$ and $i = 1, \ldots, r$, compute the inner projection,

$$\mathcal{ICT}_i[\kappa^u] = \{(x, y) \mid \forall \theta \in \mathcal{I}_u : (x, y, \theta) \in \mathcal{CT}_i[\kappa^u]\} \tag{4.3}$$

and generate bounded rectangloid approximation of $\mathcal{ICT}_i[\kappa^u] \times \kappa^u$. Then, for $\forall q \in \mathcal{RB}_j[\kappa^u]$ and $\forall \theta \in \mathcal{I}_u$, $\mathcal{A}$ avoids collisions with $\mathcal{B}_j$. And, for $\forall q \in \mathcal{RT}_i[\kappa^u]$ and $\forall \theta \in \mathcal{I}_u$, $\mathcal{S}$ can make measurements from $\mathcal{T}_i$.

3. Obtain void cells decomposition, $\mathcal{K}_{void}$: For every $u = 1, \ldots, \nu$, generate a rectangloid decomposition $\mathcal{K}^u_{void}$ of the void configuration space,

$$\mathcal{C}^u_{void} \;=\; \kappa^u \;\setminus\; \{\bigcup_{j=1}^{n} \mathcal{RB}_j[\kappa^u] \;\cup\; \bigcup_{i=1}^{r} \mathcal{R}'\mathcal{T}_i[\kappa^u]\} \tag{4.4}$$

Since $\mathcal{C}^u_{void}$ is the complement of a union of rectangloids within a rectangloid, it can easily be decomposed as a union of rectangloids. Then, let $\mathcal{K}_{void} = \cup_{u=1}^{\nu} \mathcal{K}^u_{void}$.

4. Obtain observation cells decomposition, $\mathcal{K}_z$: Let the free observation space be defined as,

$$\begin{aligned} \mathcal{C}^u_z \;&=\; \bigcup_{i=1}^{r} \mathcal{R}'\mathcal{T}_i[\kappa^u] \;\setminus\; \bigcup_{j=1}^{n} \mathcal{RB}_j[\kappa^u] \\ &=\; \bigcup_{i=1}^{r} \{\mathcal{R}'\mathcal{T}_i[\kappa^u] \;\setminus\; \bigcup_{j=1}^{n} \mathcal{RB}_j[\kappa^u]\} \equiv \bigcup_{i=1}^{r} \{\mathcal{C}^u_{z,i}\} \end{aligned} \tag{4.5}$$

For every $u = 1, \ldots, \nu$ and $i = 1, \ldots, r$, generate a rectangloid decomposition of $\mathcal{C}^u_{z,i} \setminus \bigcup_{l \neq i} \mathcal{C}^u_{z,l}$, containing cells from which only one target is observable. For every $u = 1, \ldots, \nu$, generate a rectangloid decomposition of $\bigcup_{i=1}^{r} \{\mathcal{C}^u_{z,i} \cap \bigcup_{l \neq i} \mathcal{C}^u_{z,l}\}$, containing cells from which two or more targets are observable. Then, the union of both cell decompositions constitutes the set of observation cells, $\mathcal{K}_z$.

The above decomposition is not significantly harder than a decomposition involving only obstacles. Step 4 can be carried out by using Boolean operations of rectangloids , and each rectangloid decomposition is polynomial in the number of vertices. After the above steps are completed, the entire cell decomposition $\mathcal{K} = \mathcal{K}_{void} \cup \mathcal{K}_z$, is used to obtain the following graph:

Definition 4.1.7 (Connectivity Graph with Observations) *A connectivity graph with observations, $\mathcal{G}$, is a non-directed graph where the nodes represent either an observation cell or a void cell, and two nodes in $\mathcal{G}$ are connected by an arc if and only if the corresponding cells are adjacent.*

Therefore, a connectivity graph with observations differs from a classical connectivity graph in that each observation cell is labeled and has a corresponding index set of allowable measurements (from step 4), i.e., the index set of $Z(\bar{\kappa})$. As in classical cell decomposition methods, the cost associated with moving between any two cells in the connectivity graph is attached to the arc between their corresponding nodes. In this dissertation, the motion cost between any two nodes κ_i and κ_j representing void or observation cells, and connected by an undirected arc (κ_i, κ_j) in G, is given by the following Euclidean distance in $\mathcal{C}$,

$$D(\kappa_i, \kappa_j) \equiv \max ||\mathcal{A}(\bar{q}_i) - \mathcal{A}(\bar{q}_j)|| = d_{ij} = d_{ji}. \tag{4.6}$$

taken from [49], where $\bar{q}_i$ denotes the geometric centroid of the cell κ_i.

As an example, the approximate decomposition and connectivity graph obtained for the workspace in Fig. 4.2 are illustrated in Figs. 4.3 and 4.4, respectively.

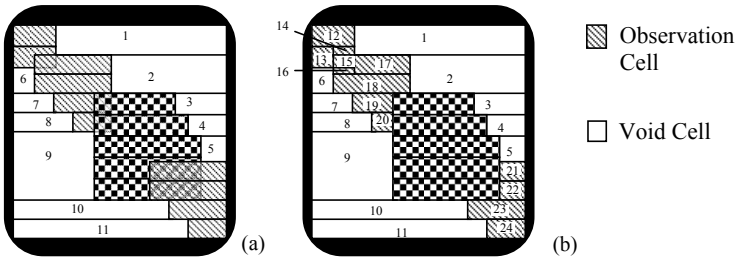


Figure 4.3: Approximate rectangloid decomposition of $\mathcal{W}$, in Fig. 4.2, into void (a) and observation (b) cells, obtained from steps (3) and (4), respectively.

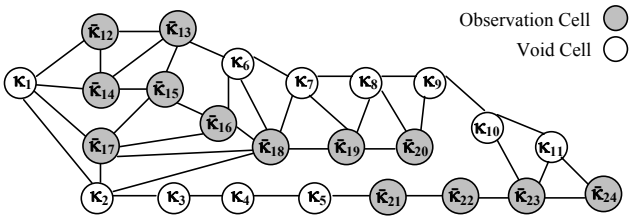


Figure 4.4: Connectivity graph obtained from the approximate rectangloid decomposition in Fig. 4.3, with observation cells labeled in grey.

4.2 Pruned Connectivity and Decision Trees

Through the connectivity graph $\mathcal{G}$ presented above, the robot and field of view geometries can be represented as a point in the configuration space. Due to the stochastic nature of the sensor measurements, the benefit associated with each observation cell cannot be represented by a deterministic function value, like the motion cost (4.6). Instead, a pre-posterior optimal policy that maximizes the expected observation profit (2.2) can be obtained by pruning $\mathcal{G}$ and transforming it into a decision tree DT. As an intermediate step, using the methodology presented in [46], $\mathcal{G}$ is pruned and represented by a connectivity tree T_r that contains a subset of feasible paths, including the one leading to the overall optimal policy.

Suppose at time t_k, the robotic sensor is in a configuration $q \in \kappa_i$. Then, if only the adjacency relations in $\mathcal{G}$ are taken into consideration, the number of cells that can be visited at t_{k+1} grows exponentially with k. However, if the objective of minimizing the distance metric (4.6) in configuration space is taken into account, the connectivity tree can be pruned at every time step, t_k, thereby eliminating a significant number of sub-optimal channels based on the principle of optimality [47]. We adopt the following definition, introduced in [46]:

Definition 4.2.1 (Connectivity Tree) *The connectivity tree T_r associated with a connectivity graph with observations, $\mathcal{G}$, and two cells containing the initial and final*

robot configurations, $\kappa_0 \ni q_0$ and $\kappa_f \ni q_f$, is a tree graph with κ_0 as the root and κ_f as the leaves. The nodes represent standard or observation cells, and an additive cost metric d is attached to each arc. A branch from the root to a leaf represents a channel *or sequence of cells joining q_0 and q_f, with the following properties:*

- *Two branches are said to be* information equivalent *if they join the same cells, κ_i and κ_l, and contain the same set of observation cells, regardless of the order.*
- *Two branches that are information equivalent can co-exist in T_r if and only if they represent the same channel.*
- *A branch in T_r connecting any two cells κ_i and κ_l has the smallest overall cost of any other information-equivalent branch in $\mathcal{G}$.*

The pruning algorithm presented in Appendix C is a type of label-correcting algorithm [50]- [51] which guarantees that T_r contains the path in $\mathcal{G}$ with the minimum cumulative cost between κ_0 and κ_f, as well as a subset of paths in $\mathcal{G}$ that are *not* information equivalent. Each of these paths enables a different subset of measurements with minimum motion cost. A proof of the properties of the resulting connectivity tree is provided in Appendix D. It is demonstrated in Section 3.3 that the order of the observations does not change the total expected observation benefit, as defined in Section 3.3. Therefore, by obtaining all paths that are not information equivalent in T_r, the observation profit can be excluded by the above transformation $\mathcal{G} \Rightarrow T_r$ without eliminating any candidate solutions to the optimal policy σ^*. As an illustrative example, consider the connectivity graph in Fig. 4.4, with κ_0 and κ_f as labeled in the figure. The corresponding connectivity tree obtained by the pruning algorithm is shown in Fig. 4.5. A branch in T_r represents a channel or sequence of cells, and a corresponding free path in configuration space for the robot $\mathcal{A}$.

In this section, a methodology is presented for representing the treasure hunt

problem (Problem 1) by means of a pruned decision tree DT, obtained by the transformations $\mathcal{G} \Rightarrow T_r \Rightarrow DT$. Decision trees can be used to represent discrete-time dynamic processes (see [33, Section4.4] for a comprehensive review). Decision and chance nodes comprise nonleaf nodes, and the leaves are the utility nodes. After a decision tree is obtained from T_r, it can be easily searched for the optimal policy by maximizing the total reward function (2.2). The obtained optimal solution is a nonmyopic global optimal solution, because the decision tree theoretically contains all candidate policies from κ_0 to κ_f and the total reward function of a policy is conditional on all future observations along the sensor path induced by the according policy.

The decision tree DT in this dissertation is a tuple $\{U_C, U_D, R, A\}$ with κ_0 as the root and the values of the reward function $R : \Omega(\beta) \rightarrow \mathbb{R}$ as the leaves. Ω is the domain of the chance and decision nodes, U_C and U_D, defined over a branch β in DT. Each branch represents a feasible sequence of cells and decisions, $\beta = \{x(t_k), u(t_k) \mid k = 0, \ldots, f\}$, and contains a channel τ connecting κ_0 to κ_f in $\mathcal{G}$. DT is constructed from T_r such that for $\forall \tau \in T_r$, $\exists \beta \in DT$ such that $\tau \subset \beta$. Branches in DT connect κ_0 to the leaves through the set of directed arcs A. The decision tree representation of the treasure hunt problem, DT, is obtained from T_r by the following assignments:

- Let the chance node $x(t_k) \in U_C$ in a branch β denote a cell $\kappa_i \in \mathcal{G}$ that the robotic sensor can visit at time t_k, such that $\kappa_i \in \tau \subset \beta$. Since $x(t_k)$ only has one instantiation κ_i, it implies the action decision to move to κ_i along τ.

- Let the test-decision node $u(t_k) \in U_D$, with $x(t_k) \prec u(t_k)$ in β, denote the set of admissible test-decisions. For example, in Fig. 4.6, if the cardinality of the set of observations $Z(\kappa_i)$ at time t_k $|Z(\kappa_i)| = 1$, $\Omega(u(t_k)) = \{\vartheta_1, \vartheta_2, \vartheta_3, \vartheta_{un}\}$,

Table 4.1: Algorithm to generate decision tree DT from T_r

For every branch $\tau \in T_r$
  for $k = 0, \ldots, f$, let $x(t_k) = \kappa_i$
    if $x(t_k)$ is an observation cell, then
      Insert a test-decision node $u(t_k)$ following $x(t_k)$
      Attach the following instantiation to the arc $(u(t_k), x(t_{k+1}))$:
      $u(t_k) = \arg\max_{\vartheta}\{w_B \cdot B(t_k) - w_J \cdot J(t_k)\}, \forall \vartheta \in \Omega(u(t_k))\}$
    end; [if]
  end;[for loop]
  Add a utility node as the leave numbered with the reward of the channel τ
  Link the chance, decision and utility nodes to form a branch $\beta \in DT$
end; [for loop]
Merge all branches β so that they only diverge from the very first different nodes in DT

where $\vartheta_j, j = 1, \ldots, 3$, denotes the test-decision to observe the target with j^{th} sensor mode and ϑ_{un} denotes the decision of not performing any measurements.

- Let arc $(x_i, x_j) \in A$, a link from node x_i to x_j, denote that x_j is taken directly following x_i. An arc from a test-decision node to a chance node is labeled with the action chosen.

The algorithm to construct a decision tree DT from the connectivity tree T_r is shown in Table 4.1. The branch in DT with maximum utility gives the optimal policy σ^* of path and sensor planning from κ_0 to κ_f.

Example 4.2.2 *The pruned connectivity tree and decision tree obtained for the workspace in Fig. 4.2 and connectivity graph in Fig. 4.4 are illustrated in Figs. 4.5 and 4.6, respectively. As is seen from Figs. 4.2 and 4.3, if* $\kappa_i \in \{\bar{\kappa}_{12}, \bar{\kappa}_{13}, \bar{\kappa}_{14}\}$, $Z(\kappa_i) = \{\mathcal{M}_1\}$; *if* $\kappa_i \in \{\bar{\kappa}_{16}, \bar{\kappa}_{17}, \bar{\kappa}_{18}, \bar{\kappa}_{19}, \bar{\kappa}_{20}\}$, $Z(\kappa_i) = \{\mathcal{M}_2\}$; $Z(\bar{\kappa}_{15}) = \{\mathcal{M}_1, \mathcal{M}_2\}$; *if* $\kappa_i \in \{\bar{\kappa}_{21}, \bar{\kappa}_{22}, \bar{\kappa}_{23}\}$, $Z(\kappa_i) = \{\mathcal{M}_3\}$. *It is assumed in this example that the expected information benefit of* $\mathcal{M}_1, \mathcal{M}_2$ *and* $\mathcal{M}_3$ *over the test decision domain* $\{\vartheta_1, \vartheta_2, \vartheta_3, \vartheta_{un}\}$

orderly is $\{0.25,\ 0.2,\ 0.4,\ 0\}$, $\{0.03,\ 0.02,\ 0.01,\ 0\}$ *and* $\{0.15,\ 0.25,\ 0.1,\ 0\}$, *respectively. In the algorithm shown in Table 4.1,* $w_B = 20, w_J = 1$. $J(t_k)$ *is set to equal the times of sensor taking measurements; otherwise,* $J(t_k) = 0$. *Once one measurement* $\mathcal{M}_i$ *is taken, the information benefit of taking another* $\mathcal{M}_i$ *along the same branch will become* 0 *and thus the corresponding test decision is chosen to be "not performing any tests", i.e.,* ϑ_{un}. *For example, in the first branch in Fig. 4.6,* $u(t_6) = \vartheta_{un}$ *at cell* $\bar{\kappa}_{22}$ *since* $\mathcal{M}_3$ *has been taken at* $\bar{\kappa}_{21}$. *In Fig. 4.6, the branch with the total utility of* 0.2 *represents the optimal policy* σ^* *from* $\kappa_0 = \kappa_1$ *to* $\kappa_f = \kappa_{11}$.

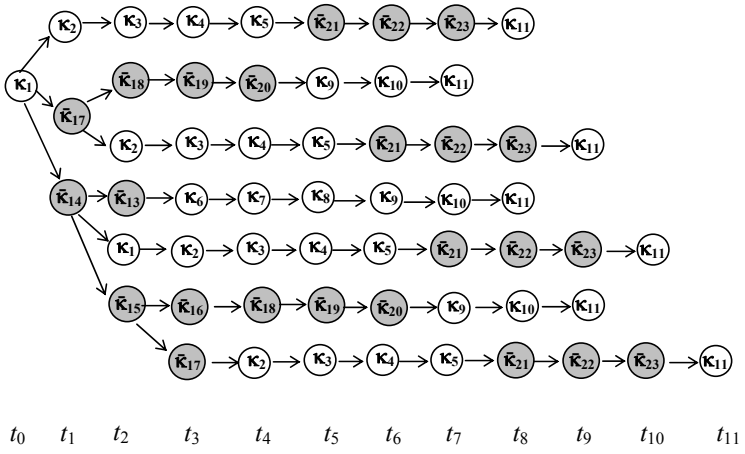


Figure 4.5: Connectivity tree T_r obtained from the connectivity graph in Fig. 4.4 via pruning algorithm.

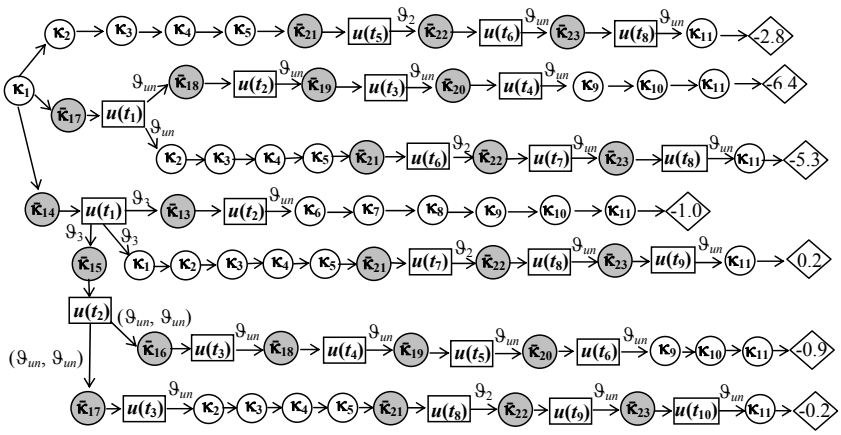


Figure 4.6: Decision tree DT obtained from the connectivity tree in Fig. 4.5.

Chapter 5

Application I: Optimal Strategies in the Board Game of CLUE®

The board game of CLUE® constitutes an excellent benchmark example. While its rules and objectives can be easily understood, and are already well known by most, this game exhibits all the basic challenges of the treasure hunt problem.

A key aspect of the game is that in order to make a suggestion involving a particular room and collect evidence about the corresponding card, the player's pawn must be inside the room. Hence, the type of evidence that may be collected depends on the position of the pawn.

Bayesian models of uncertainty have been used for game playing in [52], where the expected payoffs were obtained from a probabilistic evaluation function learned from data. Although there exist several computer version of CLUE®, to our knowledge this is the first mathematical treatment of CLUE® game strategies. The influence diagram representation of CLUE® [46], or decision tree obtained by the methodology in Section 4.2, is used to compute optimal game strategies, including moving the pawn and making suggestions. The strategies are tested through an interactive simulation of CLUE® that allows human players to confront the computer, which is referred to as intelligent computer player (ICP). In each trial, one human player (HP) confronts the ICP and a constrained satisfaction player (CSP) as shown in Section 5.3.6.

5.1 Rules of the Game

The board game of CLUE® is a popular detective game where the objective of each player is to determine the guilty suspect, weapon, and room of an imaginary murder that takes place in the CLUE® mansion (Fig. 5.1). In this mansion, there are nine rooms: the dining room (r_d), the library (r_l), the billiard room (r_b), the hall (r_h), the kitchen (r_k), the lounge (r_g), the ballroom (r_a), the study (r_s), and the conservatory (r_c). The players each choose a pawn representing one of the six potential suspects: Colonel Mustard (s_m), Miss Scarlett (s_t), Professor Plum (s_p), Mr. Green (s_g), Mrs. White (s_w), and Mrs. Peacock (s_k). In this dissertation, it is assumed that there are three players: the human player, the ICP, and the RCP. The imaginary murder can take place by one of six weapons: knife (w_k), rope (w_r), candlestick (w_c), lead pipe (w_p), revolver (w_v), and wrench (w_h). Each item in the game is represented by an illustrated card, such that there is a total of twenty-one cards in the deck.

At the onset of the game one card from each item category (room, suspect, and weapon) is randomly selected, removed from the deck and hidden in an envelope representing the murder's guilty suspect, room, and weapon for the remainder of the game. The remaining cards are dealt to the players who subsequently eliminate their own cards from the list of possible hidden items. The players move their pawns about the mansion by rolling the die, and when they enter a particular room, say r_i, they can make a suggestion that involves r_i and any item from the suspect and weapon category. It is by making suggestions that the players collect information about the other players' cards and, consequently, can infer the hidden cards in the envelope.

There are three ways for a pawn to enter a room: (i) a door illustrated on the game board (Fig. 5.1), (ii) a secret passage connecting opposite corners (Fig. 5.1), or (iii) being suggested as the murder suspect by one of the other players. Upon

entering a room, a player p_i must make a suggestion involving the potential room, suspect, and weapon. The next player in the turns' rotation, p_{i+1}, must refute the suggestion by showing one these cards, if possible, and otherwise state that he has none of the suggested cards. In the latter case, the next player, p_{i+2}, must attempt to refute the suggestion. Then, the turn goes to the next player (in this case, p_{i+1}). When neither p_{i+1} nor p_{i+2} are able to refute it, it can be inferred that the suggested cards are the hidden ones, or that the player p_i has suggested one of her own cards. All of these possibilities, and even bluffs, are allowed in the CLUE® simulation.

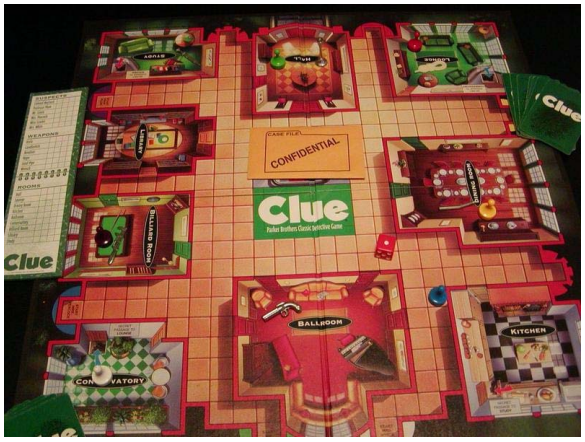


Figure 5.1: CLUE® mansion and game pieces. CLUE® & ©2006 Hasbro, Inc. Used with permission.

5.2 Interactive CLUE® Simulation

The hypothesis-driven path planning technique developed in this dissertation is tested by means of an interactive simulation of the game of CLUE® that allows experienced human players to play against the ICP, through the MATLAB Graphical User Interface (GUI) Toolbox. The human player assigns pawns to the three players, the HP, the ICP, and the RCP, and the simulation deals the cards randomly to the players. At any time the human player can see his cards on the screen at the command of

a button. The die is replicated by a random number generator that produces an integer between one and six. The mansion is illustrated on a virtual board where the players can move the pawns within the distance given by the die roll, and can enter the various rooms through the doors or secret passages, as in the real game. At each turn, a player must (1) roll the virtual die; (2) move the pawn on the virtual board by a distance no greater than the die roll and, possibly, enter a room by clicking on the desired location; (3) make a suggestion; (4) make an accusation or transfer the turn to the next player; all in this sequence. The human player makes a suggestion (or an accusation) by means of pull-down menus containing all twenty-one items, and one of the computer players refutes the suggestion by displaying one of its cards on the screen. The computer players make a suggestion by displaying the three desired cards on the screen. If a computer player must refute this suggestion a warning is displayed on the screen to the HP, but the value of the card is exchanged only between the computer players as data in the MATLAB environment. If it is the turn of the HP to refute the computer's suggestion, then an appropriate pull-down menu is displayed such that the HP can choose which card to "show" to the computer.

Typically, human players take notes during the game to record the outcomes of the suggestions, while the computer players record and access information through the MATLAB environment. Human players can see their cards on the screen at any time during their turn, and their actions are observed by the computer through MATLAB interfaces.

5.3 Methodology and Results

The moving pawns are viewed as point robots navigating around the board. The suggestions in the game of CLUE® are viewed as measurements that depend on the room (or target), and whose outcomes are are unknown *a priori*. Based on a

probabilistic argument and on the evidence obtained during the game, it is possible to develop a pre-posterior strategy for navigating the board optimally and gather information about the hidden cards. A game strategy consists of the test decision on whether to enter a room and make a suggestion, and of action decision on how to move the pawn inside the game board. The benefit of reaching a room (or target) is the reduction in uncertainty that is expected from the corresponding suggestion. The costs of reaching a room and entering it are the distance traveled and the turn missed for making a suggestion, respectively.

5.3.1 Information Reward Function in the CLUE®

In hypothesis-driven decision problems the benefit of performing one or more tests or measurements $m_i, \ldots, m_j \in M$ is to decrease the uncertainty of the hypothesis variable y. Since the outcome of the tests is unknown *a priori*, the benefit of observation, $B(t_k)$,is defined over the posterior probability distribution $P(y|m_i, \ldots, m_j)$. Information entropy, conditional entropy and mutual information (or EER) is defined in eq. 3.2, 3.3 and 3.4. Then, the following result provides an appropriate function for the benefit of observation.

Theorem 5.3.1 *Let $Z_T = \{z_1, z_2, \ldots, z_{f-1}\}$ be a sequence of measurements about an hypothesis variable y that are performed through a Markov process defined over a set of decision epochs $T = \{t_1, t_2, \ldots, t_f\}$. Then, the incremental entropy,*

$$\begin{aligned} \Delta H(t_k) &\equiv H(t_{k-1}) - H(t_k) \\ &= H(y|z_{k-1}, z_{k-2}, \ldots, z_1) - H(y|z_k, z_{k-1}, z_{k-2}, \ldots, z_1) \\ &= I(y;\ z_k \mid z_{k-1}, z_{k-2}, \ldots, z_1) \end{aligned} \tag{5.1}$$

is a conditional mutual information, and represents the reduction in uncertainty in y that is incurred at time t_k, when an additional measurement z_k is performed. The

incremental entropy is a reward function that is additive over time. Hence, the total measurement benefit,

$$B(t_f) = \sum_{k=1}^{f-1} \Delta H(t_k) = I(y;\ z_1, z_2, \ldots, z_{f-1}) \tag{5.2}$$

is the reduction in uncertainty brought about by all measurements in Z_T.

See Appendix E for the proof. Also, the following result follows from the above theorem, which is useful in obtaining a decision policy that leads to an optimal measurement sequence $Z_T = \{z_1, z_2, \ldots, z_{f-1}\}$ chosen from a set M (with $f < r$):

Remark 5.3.2 *The total measurement benefit at any time $t_i \in (t_0, t_f]$,*

$$B(t_i) = \sum_{k=1}^{i} \Delta H(t_k) \tag{5.3}$$

that is obtained from a sequence of tests whose individual outcomes are independent of time, is independent of the order in which the tests are performed.

A proof is provided in Appendix F. The same result applies to the cost of measurement, J, provided it is an additive function. It follows that, as shown in Appendix D, the pruning algorithm can eliminate information-equivalent branches based solely on distance, without eliminating solutions that are optimal with respect to the total measurement profit (2.2). As shown in (5.3), at any time t_k incremental entropy must be computed for all combinations of measurement outcomes that may be obtained prior to t_k. Therefore, the results in the following section are derived to demonstrate that for certain forms of the joint PMF, $P(y, M)$, the incremental entropy can be obtained efficiently using an approach that is recursive with respect to t_k.

5.3.2 Efficient Incremental Entropy Computation Over Time

Based on the formulation of the treasure hunt problem, the hypothesis variable y must be inferred from a subset of M, given $P(y, M)$. From eq. (5.1), the measurement benefit for an admissible measurement, $z_k = m_j \in M$, at time t_k is,

$$I(y;\ z_k = m_j \mid Z_{t_{k-1}}) = E_{y,m_j,z_{k-1},\ldots,z_1}\left\{\log_2 \frac{P(y, m_j \mid Z_{t_{k-1}})}{P(y \mid Z_{t_{k-1}})P(m_j|Z_{t_{k-1}})}\right\} \tag{5.4}$$

and must be computed for every admissible sequence of measurements $Z_{t_{k-1}} \equiv \{z_1, z_2, \ldots, z_{k-1}\}$, obtained before t_k. The joint probability $P(y, m_j, Z_{t_{k-1}})$ can be obtained by marginalizing the remaining measurements out of the joint probability distribution $P(y, M)$, and the posterior probabilities in eq. (5.4) can be obtained from $P(y, m_j, Z_{t_{k-1}})$ by a straightforward application of the general factorization property and Bayes rule.

Suppose the joint probability $P(y, M)$ is known in the form of the Bayesian network (BN) in Fig. 5.2.a, that represents a particular form of factorization,

$$P(y, M) = P(y) \cdot \prod_{m_j \in M} P(m_j \mid y) \tag{5.5}$$

where all of the terms are known from the BN CPTs, shown in Fig. 5.2.a. Then, by exploiting the above factorization, the incremental entropy can be computed recursively over time. At time t_0, the entropy of the hypothesis variable is $H(t_0) = H(y)$ and can be obtained from the CPT $P(y)$ in Fig. 5.2.a, as shown in eq. (3.2). If the subset of measurements M_k are obtained by the sequence of measurements Z_{t_k}, and none of the measurements are ever repeated over the time period $(t_0, t_f]$, then $H(t_k) = H(y \mid Z_{t_k})$ can be obtained from eq. (3.3), using the posterior probability,

$$P(y \mid Z_{t_k}) = \frac{P(y, Z_{t_k})}{P(Z_{t_k})} = \frac{\sum_{m_i \notin M_k} P(y, M)}{P(Z_{t_k})} = \frac{\sum_{m_i \notin M_k} \{P(y) \cdot \prod_{m_j \in M} P(m_j \mid y)\}}{P(Z_{t_k})}$$

$$= \frac{P(y) \cdot \{\prod_{m_i \in M_k} P(m_i \mid y)\} \cdot \{\sum_{m_j \notin M_k} \prod_{m_j \notin M_k} P(m_j \mid y)\}}{P(Z_{t_k})}$$

$$= \frac{P(y) \cdot \prod_{m_i \in M_k} P(m_i \mid y)}{P(Z_{t_k})} = \frac{P(z_k = m_l \mid y) \cdot P(y, Z_{t_{k-1}})}{P(Z_{t_k})} \tag{5.6}$$

and the joint probability

$$P(Z_{t_k}) = \sum_y P(y, Z_{t_k}) = \sum_y P(z_k = m_l \mid y) \cdot P(y, Z_{t_{k-1}}) \tag{5.7}$$

The last term in the numerator of eq. (5.6) equals one by the unity law of probability, and $P(z_k = m_l \mid y)$ is available from the BN CPT associated with node m_l. The above distributions are both formulated with respect to $P(y, Z_{t_{k-1}})$, which is available from the previous time step, letting $k = 0, 1, 2, \ldots, f$. Thus, the incremental entropy is computed as $\Delta H(t_k) = H(t_{k-1}) - H(t_k)$, where $H(t_{k-1})$ also is known from the previous time step. If there is more than one admissible measurement at any time $t_0 \leq t_i \leq t_k$, then the above computation must be carried out for every one of them, but can still be performed recursively. This recursive entropy computation exploits the BN factorization to gain computational efficiency compared to a brute-force marginalization of the joint probability distribution, $P(y, M)$. A similar approach also is used in the arc-reversal method and in junction-tree propagation algorithms [53].

Another case in which the incremental entropy computation can be carried out recursively is that of a BN structure that has feed-forward connections among the measurement nodes, as illustrated in Fig. 5.2.b. The measurement nodes are ordered such that if $m_i < m_j$, then there is an arc from m_i to m_j, and the joint probability including the hypothesis variables exhibits the following factorization,

$$P(y, M^o) = P(y) \cdot \prod_{j=1}^{r} P(m_j \mid y, m_{j-1}, \ldots, m_1) \tag{5.8}$$

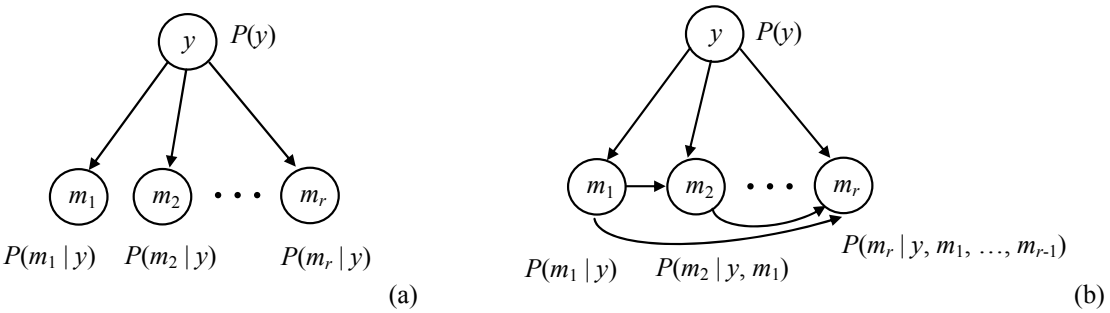


Figure 5.2: Examples of efficient Bayesian network factorization of the joint probability distribution $P(y, M)$.

where, the measurement set is now totally ordered $M^0 \equiv \{m_1, \ldots, m_r\}$. This type of BN structure may be useful in representing the probabilistic relationships between measurements with the same domain that are interchangeable and thus can be ordered according to the time at which they are performed, such that $M_k^o = \{m_1, m_2, \ldots, m_k\} = Z_{t_k}$, and $M_k^o \subset M^o$. Although the incremental entropy is computed by the same approach, the posterior probability needed to compute $H(t_k)$ is now obtained by exploiting the new BN factorization (eq. 5.8),

$$P(y \mid Z_{t_k}) = P(y \mid z_k, Z_{t_{k-1}}) = \frac{P(z_k \mid y, Z_{t_{k-1}}) \cdot P(y \mid Z_{t_{k-1}})}{P(z_k \mid Z_{t_{k-1}})} \tag{5.9}$$

and by applying the probability law $P(y \mid z_1, z_2) = P(z_2 \mid y, z_1) \cdot P(y|z_1)/P(z_2|z_1)$. The posterior probability at the denominator also can be computed recursively,

$$P(z_k \mid Z_{t_{k-1}}) = \sum_y P(y, z_k \mid Z_{t_{k-1}}) = \sum_y P(z_k \mid y, Z_{t_{k-1}}) \cdot P(y \mid Z_{t_{k-1}}),$$

as well as the joint probability table needed for computing the entropy:

$$P(y, Z_{t_k}) = P(y, z_k, Z_{t_{k-1}}) = \sum_{m_i \notin M_k^o} P(y, M^o)$$

$$= \sum_{m_j \notin M_k^o} P(y) \cdot \prod_{j=1}^{r} P(m_j \mid y, m_{j-1}, \ldots, m_1)$$

$$= P(y) \cdot \prod_{i=1}^{k} P(m_i \mid y, m_{i-1}, \ldots, m_1) \cdot \sum_{\forall m_j > m_k} \{ \prod_{j=k+1}^{r} P(m_j \mid y, m_{j-1}, \ldots, m_1) \}$$

$$= P(y) \cdot \prod_{i=1}^{k} P(m_i \mid y, m_{i-1}, \ldots, m_1) = P(z_k \mid y, Z_{t_{k-1}}) \cdot P(y, Z_{t_{k-1}}) \tag{5.10}$$

The above equation is an efficient computation that allows the entropy to be computed recursively from the probability distributions required by the previous time step, $P(y \mid Z_{t_{k-1}})$ and $P(y, Z_{t_{k-1}})$, and from the available BN CPT $P(z_k \mid y, Z_{t_{k-1}}) = P(m_k \mid y, m_{k-1}, \ldots, m_1)$.

One possible approach is to use the BNs in Fig. 5.2 to compute the probabilities needed by the benefit function (5.3) at every time slice t_k. Another approach, used in this dissertation, is to include the hypothesis variable y in the ID representation of underlying POMDP, as shown in Fig. 5.3, and to assign the probabilities obtained in this section to the hidden node $y(t_k)$. This hidden node has the same domain as y, and $\pi(y(t_k)) = \{z(t_1), \ldots, z(t_k)\}$. Since y is parent only to the utility node, the ID in Fig. 5.3 does not have a complexity problem, provided x_k is observable. If x_k is hidden, or if the domain of M is very large, then the solution of this ID may require information blocking or the use of a limited memory ID (LIMID) [54]. An example of treasure hunt problem and solution is provided in the next section. Unlike common POMDP where the utility nodes are formulated as the cost and the expected cost is minimized, the proposed ID formulates utility nodes as information benefit, which is a function of the posterior distribution of hypothesis variable $y(t_k)$, and will maximize the expected information reward. One of the interesting issues is the convergence property of the ID representation of underlying POMDP under

new information benefit formulation of utility nodes. But the convergence problem of the proposed ID is beyond the topics of this dissertation, since the ID is only used as a tool to solve the treasure hunt problem arising in the game of CLUE® and the decision tree proposed in Section 4.2 is also an effective approach. Furthermore, in the CLUE® simulations that have run, the ID never fails to produce optimal policies.

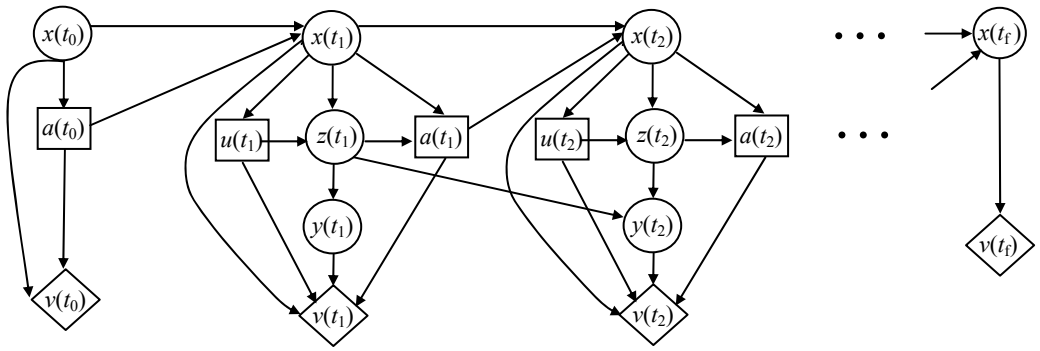


Figure 5.3: Influence diagram representation of the treasure hunt problem (the CPTs attached to y_k, $P(y \mid Z_{t_k})$, are obtained from a BNs (e.g., Fig. 5.2) by arc reversal.

5.3.3 Connectivity Tree for Navigating the CLUE® Mansion

The CLUE® mansion (Fig. 5.1) constitutes the pawns' two-dimensional workspace $\mathcal{W}$. Since the rooms can only be accessed through doors or secret passages, these locations represent the set of observation cells $\mathcal{K}_{\mathcal{T}_i}$ that enable the suggestion or measurement m_i, associated with the room $\mathcal{T}_i$. The ICP pawn can translate in the horizontal or vertical directions, moving from bin to bin in the two-dimensional grid illustrated on the game board (Fig. 5.1). Hence, the connectivity graph of this workspace can be obtained by a convex polygonal decomposition (CPD) [48] comprised of polygons that are decomposable into the existing square bins, as shown in Fig. 5.4. The corresponding CLUE® connectivity graph is illustrated in Fig. 5.5, where the nodes representing void cells are shown in white, and the nodes representing

observation cells are shown in grey (the room that is entered through these cells is shown by a dashed line). Since the pawn can only move to adjacent bins according to the number rolled, the appropriate workspace metric is the Manhattan distance or taxicab metric [55], that is,

$$d_{ij} = D(\kappa_i, \kappa_j) = ||c_{x_i} - c_{x_j}|| + ||c_{y_i} - c_{y_j}|| = d_{ji} \tag{5.11}$$

where, $||\cdot||$ denotes the L_2-norm in units of $bins$, and (c_{x_i}, c_{y_i}) are the xy-coordinates of the geometric centroid of the i^{th} cell.

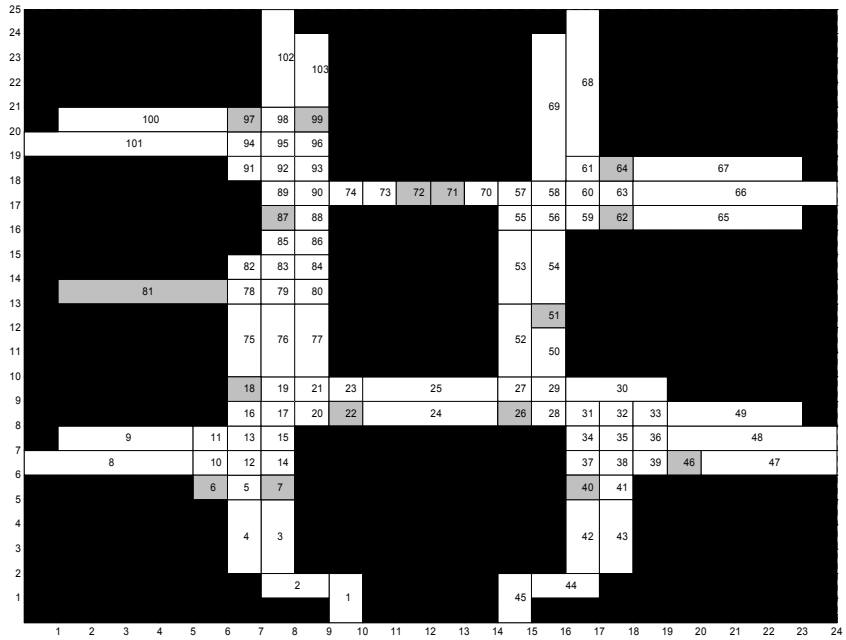


Figure 5.4: Convex polygonal decomposition (CPD) of the CLUE® workspace, where void cells are shown in white, observation cells in grey, and obstacles in black.

The connectivity tree obtained by the pruning algorithm is shown in Table 5.1, for $\kappa_0 = \kappa_3$ and $\kappa_f = \kappa_{51}$. Every time κ_0 or κ_f change, a new connectivity tree must be obtained from $\mathcal{G}$. Although the pawn's location typically is given by the optimal strategy, the pawn may be moved to a room by a player suggesting the ICP pawn's character as the potential suspect. In this case, a new connectivity tree is

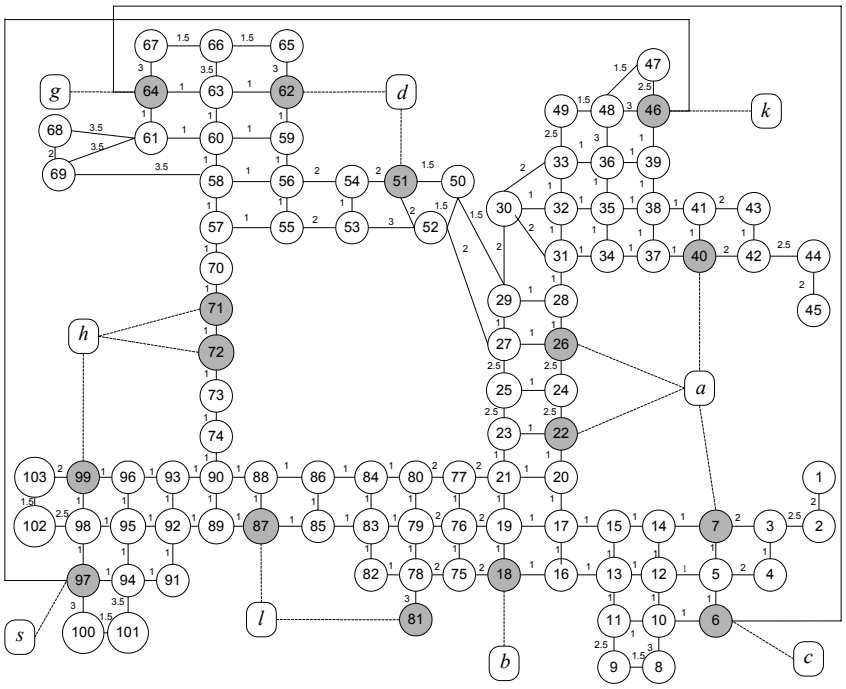


Figure 5.5: Connectivity graph with observations for CLUE®, corresponding to the CPD in Fig. 5.4, with dashed lines indicating the room that can be entered through each observation cell (adjacent to the room door).

Table 5.1: Connectivity tree for $q_0 \in \kappa_3$ and $q_f \in \kappa_{51}$ (arcs and costs are as shown in Fig. 5.5).

t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}
1	2	3	4	a	6	a	6	c	g	a	g	7	d
t_{14}	t_{15}	t_{16}	7	5	7	c	7	5	b	c	b	14	14
15	b	b	t_{17}	14	15	5	15	14	a	7	a	15	15
17	a	18	b	t_{18}	22	14	16	17	6	14	7	17	16
18	17	19	a	18	26	17	19	18	7	17	14	18	17
19	18	20	18	19	t_{19}	23	20	21	15	18	15	21	19
20	19	21	19	21	19	24	22	22	19	21	19	22	20
21	20	22	21	22	21	27	25	23	20	22	20	23	21
22	21	23	22	23	23	t_{20}	26	24	23	25	22	24	23
23	22	24	23	24	24	21	52	27	24	26	23	25	24
25	23	25	24	25	25	23	t_{21}	51	25	27	24	26	25
26	24	26	25	26	26	25	23	t_{22}	26	51	26	27	26
27	25	27	26	27	27	26	25	25	51	64	27	51	27
51	27	51	27	51	51	27	27	27	52		51	52	51
52	51	52	51	52	52	51	51	51	t_{23}	t_{24}	52	60	52
56	52	56	52			52	52	52	27	51	61	62	58
60	54		54						51	52	63	64	59
62	58								52		t_{25}	69	61
69	59										51		63

generated using the new initial position κ_0. Also, a new connectivity tree is required when the desired final position κ_f changes as a consequence of the evidence obtained during the game. In CLUE®, κ_f is nearest observation cell adjacent to the room with maximum probability of being the hidden one. The joint probability distribution over the CLUE® cards and the insertion of evidence are modeled by a BN, as shown in the next section.

5.3.4 CLUE® Bayesian Network

The inference process required to estimate the hidden CLUE® cards representing the murder's weapon, room, and guilty suspect, is automated by means of a Bayesian network that models the relationships between the game cards. Assuming there are three players in the game, the eighteen cards remaining in deck can be equally distributed into groups of six cards per player. Every card in the deck is represented by one node in the CLUE® BN. The hidden room (y^r), weapon (y^w), and suspect (y^s) cards are represented by three nodes whose domain equals the domain of the cards' respective categories, i.e.: $\Omega(R) = \{r_d, r_l, r_b, r_h, r_k, r_g, r_a, r_s, r_c\}$, $\Omega(S) = \{s_m, s_t, s_p, s_g, s_w, s_k\}$, and $\Omega(W) = \{w_k, w_r, w_c, w_p, w_v, w_h\}$. Every time a card is dealt, its value influences those of the cards that are dealt subsequently. The hidden cards influence only the cards in the same category. However, since the categories of the players' cards are unknown, a general BN accounting for all possible relationships has the structure shown in Fig. 5.6, where C_{ji} denotes the j^{th} card of i^{th} player, and an arc is placed between two cards $C_{ji} \rightarrow C_{lk}$ when $k > i$, or when $k = i$ and $l > j$. Also, every card C_{ji} has twenty-one possible outcomes, i.e., $\Omega(C_{ji}) = \Omega(R) \cup \Omega(S) \cup \Omega(W)$. The BN in Fig. 5.6 has a complexity problem, because the CPT attached to the last node C_{63} is of order $\mathcal{O}(10^{25})$.

A tractable CLUE® BN structure is obtained by introducing an assumption on

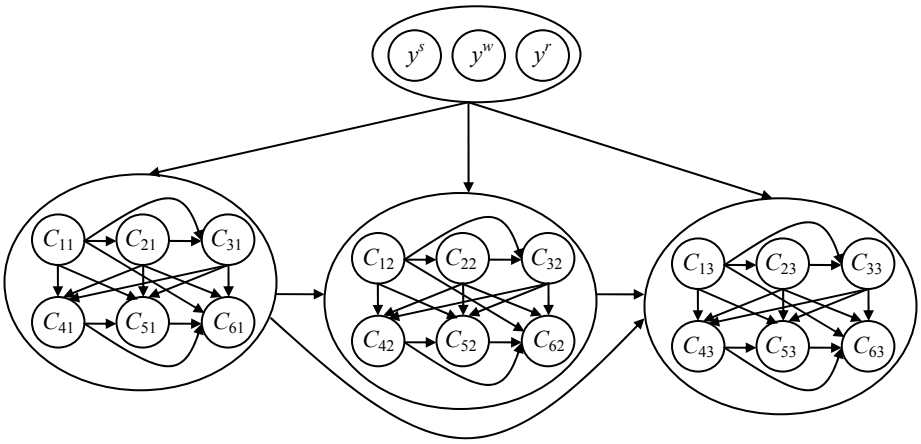


Figure 5.6: Structure of the exact BN model for the CLUE® cards, where an arc between two node clusters indicates connections among all the nodes in the clusters in the direction shown.

the categories of the cards that are dealt to each player. The cards are dealt uniformly among the three players, such that: (i) p_1 has two suspect cards, one weapon card, and three room cards; (ii) p_2 has one suspect card, two weapon cards, and three room cards; and, (iii) p_3 has two suspect cards, two weapon cards, and two room cards. The ICP is chosen to always be p_3 who has the disadvantage of having less room cards than the other players. p_1 and p_2 can be either human or random computer players. Also, the ICP always knows its own cards, therefore they are excluded from the CLUE® BN, and their values are eliminated from the BN domain, which is created at the onset of game. Let C^s_{ji}, C^w_{ji}, and C^r_{ji} denote the j^{th} suspect, weapon, and room cards, respectively, that each belong to the i^{th} player, with $i = 1, 2$. After the cards are dealt by the simulation, the BN domain is determined by the following set operations,

$$\begin{aligned}
\Omega(y^s) &= \Omega(C^s_{ji}) = \Omega(\bar{S}) \equiv \Omega(S) \setminus \Omega(C^s_{l3}) = \{s_\imath : \imath = 1, \ldots, 4\} \\
\Omega(y^w) &= \Omega(C^w_{ji}) = \Omega(\bar{W}) \equiv \Omega(W) \setminus \Omega(C^w_{l3}) = \{w_\jmath : \jmath = 1, \ldots, 4\} \\
\Omega(y^r) &= \Omega(C^r_{ji}) = \Omega(\bar{R}) \equiv \Omega(R) \setminus \Omega(C^r_{l3}) = \{r_\iota : \iota = 1, \ldots, 7\}
\end{aligned}$$

with $i, l = 1, 2$. Since cards of different categories are conditionally independent, with the above assumption when a card is hidden or dealt to a player it only influences the cards that are dealt subsequently and are in the same category. Thus, the relationships between the hidden cards and the cards dealt to p_1 and p_2 can be modeled by the CLUE® BN in Fig. 5.7, for which inference is feasible.

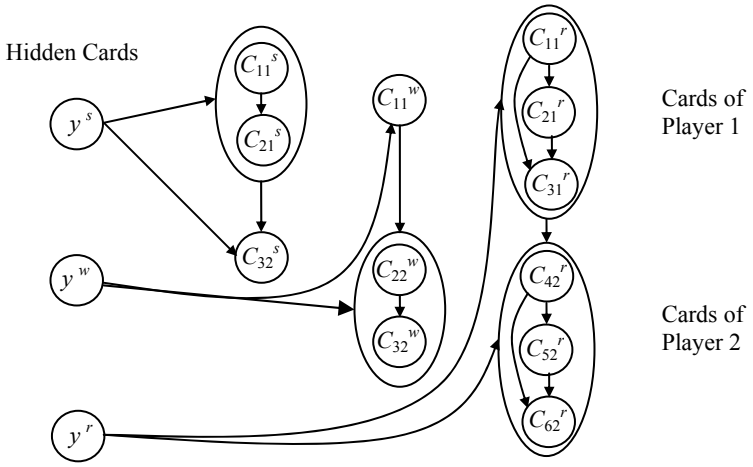


Figure 5.7: Structure of approximate BN model for the CLUE® cards, where it is assumed that Player 1 is dealt two suspect cards, one weapon card, and three room cards, and Player 2 is dealt one suspect card, two weapon cards, and three room cards.

The CPTs of the CLUE® BN (Fig. 5.7) are obtained by inspection: cards with the same domain may not acquire the same instantiation, and cards with different domains are independent. Let a binary variable υ be defined over a set of finite and discrete nodes, $X = \{x_1, \ldots, x_n\}$, that all have the same domain of n mutually exclusive state values $\Omega(x) = \{x^1, \ldots, x^n\}$, such that,

$$\upsilon(X) = \left(\prod_{l} \prod_{k \neq l} (1 - \delta_{ij}) \, : \, x_l = x^i, \; x_k = x^j, \; \forall \, (x_l, x_k) \in X; \; x^i, x^j \in \Omega(x) \right). \quad (5.12)$$

Then, the CPTs of the CLUE® BN shown in Fig. 5.7 are given by,

$$P(C_{ji}^{\ell} \mid pa(C_{ji}^{\ell})) = \frac{\upsilon(C_{ji}^{\ell} \cup pa(C_{ji}^{\ell}))}{|\Omega(C_{ji}^{\ell})| - |pa(C_{ji}^{\ell})|} \tag{5.13}$$

and,

$$P(y^{\ell} = \ell_j) = \frac{1}{|\Omega(y^{\ell})|}, \ \ \forall \ell_j \in \Omega(y^{\ell}) \tag{5.14}$$

The symbol $|\cdot|$ denotes the cardinality of a set, and ℓ denotes the card category. Hence, the above equations hold for $\ell = s, w, r$. This BN is used by the ICP to make an intelligent suggestion any time its pawn enters a room in the CLUE® mansion. It also is used to obtain the joint probability distribution over the hypothesis variables and the tests that can be performed during the game, as illustrated in Section 5.3.6. The following section illustrates how evidence is computed and entered in the CLUE® BN, based on the replies provided by the other players during the game.

5.3.5 Evidence Tables Construction and Update

New evidence can become available during every player's turn, when suggestions are proven or refuted by the other two players. The simplified CLUE® BN in Fig. 5.7 consists of three BNs, one for each card category. Therefore, the game evidence can be organized into three independent tables, E^s, E^w, and E^r, that are used to store hard and soft evidence about the respective BN nodes. *Hard evidence* refers to perfect knowledge of a node's instantiation. If the probability distribution of a node over its possible values is known, it is referred to as *soft evidence* (*se*), and is represented by a known probability distribution function $Q(\cdot)$. Jeffrey's rule is used as a mechanism for updating soft evidence in a BN with an unknown node x, given

soft evidence about its children $ch(x)$:

$$P(x \mid se) = \sum_{ch(x)} P(x \mid ch(x))\, P(ch(x) \mid se) = \sum_{ch(x)} P(x \mid ch(x))\, Q(ch(x)) \qquad (5.15)$$

Hard evidence can be viewed as a special type of soft evidence, where a value of one is assigned to the instantiated value, and all the other values have a zero probability in Q.

The evidence tables contain the latest soft or hard evidence about the players' cards, that are the children of the hidden hypothesis variable y^s, y^w, and y^r to be inferred. Let E^ℓ denoted a $(N \times M)$ matrix of known probabilities for all children of the hypothesis variable y^ℓ, such that every row in E^ℓ contains $Q(C^\ell_{ji})$, for all $C^\ell_{ji} \in ch(y^\ell)$ (as shown in Fig. 5.7), where $N = |ch(y^\ell)|$ and $M = |\Omega(y^\ell)|$. At the onset of the game, all cards have equal probability to assume any of the values in their respective domain. Hence, the evidence tables are initialized according to the following equation,

$$E^\ell = \{e^\ell_{kl}\} = \left\{ \frac{1}{|\Omega(y^\ell)|} \right\}, \text{ at } t_0 \qquad (5.16)$$

where e^ℓ_{kl} represents the element in the k^{th} row and l^{th} column of E^ℓ. After t_0, the evidence tables are updated with the evidence obtained through the players' suggestions, based on the following rules:

1. By the unity law, all probabilities in the same row must sum to one,

$$\sum_{l=1}^{M} e^\ell_{kl} = 1$$

   where, $M = |\Omega(y^\ell)|$.

2. Hard evidence negating an instantiation is constant over time, therefore,

$$\text{if } C^\ell_{ji} \neq \ell_\iota \text{ at } t_\iota, \text{ then } Q(C^\ell_{ji} = \ell_\iota) = 0, \ \forall t > t_\iota.$$

3. Hard evidence supporting an instantiation is constant over time and also negates the instantiation for all other cards in the same category. Thus,

$$\text{if } C^{\ell}_{ni} = \ell_m \in \Omega(y^{\ell}) \text{ at } t_\iota, \text{ then } e^{\ell}_{nm} = 1, e^{\ell}_{nl} = e^{\ell}_{km} = 0,$$

for, $\forall k \neq n$, $\forall l \neq m$, and $\forall t > t_\iota$.

The evidence tables are updated at every player's turn, using a rule-base system derived from the CLUE® game rules. Let the set of three instantiations $\mathcal{G} = \{s_\mathcal{G}, w_\mathcal{G}, r_\mathcal{G}\}$ denote the suggestion of a player whose pawn is in room $r_\mathcal{G}$, where $s_\mathcal{G} \in \Omega(S)$, $w_\mathcal{G} \in \Omega(W)$, and $r_\mathcal{G} \in \Omega(R)$. There are two cases that allow the ICP to obtain soft evidence from the suggestion of another player: (a) one or more players are not able to refute the suggestion of a player p_i, with $i = 1, 2, 3$, revealing that they do not possess any of the three cards; or, (b) a player refutes the suggestion made by a player other than the ICP. In these cases, the ICP can observe $\mathcal{G}$, but not the actual instantiation in $\mathcal{G}$ that was used to refute it. The information content of the latter observation depends on the hard evidence that was previously available. Hard evidence is obtained only when the ICP makes a suggestion that is refuted by one of the other players who shows one card to the ICP. All evidence is stored in the evidence tables, but the type of evidence determines how the tables are updated. The evidence tables update consists of two steps: I. updating the probabilities, and II. guaranteeing that the new probabilities satisfy rules (1)-(3).

I. The evidence tables probabilities are updated to reflect the evidence observed from the latest turn, t_ι, as follows:

   a) *Soft Evidence:* Let $\mathcal{G}$ denote a suggestion by a player p_i that cannot be refuted by at least one other player. Then, for any p_k ($k \neq i$, $k \neq 3$) that

cannot refute $\mathcal{G}$, set

$$e^{\ell}_{jm} = 0, \text{ for } \forall \ell_{\mathcal{G}} \in \mathcal{G}, \text{ and } \forall\, j:\ C^{\ell}_{jk} \in ch(y^{\ell}) \tag{5.17}$$

where $\ell_{\mathcal{G}} = \ell_m \in \Omega(y^{\ell})$.

b) *Soft Evidence:* Let $\mathcal{G}$ denote a suggestion by a player p_i that is refuted by p_k $(i, k \neq 3,\ i \neq k)$, by showing a card $C_{\mathcal{G}}$ unknown to the ICP, but such that $\Omega(C_{\mathcal{G}}) = \mathcal{G}$. Let $\phi^{\ell} = \Omega(y^{\ell}) \setminus \{\ell_m : e^{\ell}_{nm} = 1,\ \forall C^{\ell}_{ni} \in ch(y^{\ell})\}$ denote the set of free instantiations for cards of the ℓ category, based on E^{ℓ}_{old}. The set $\phi_{\mathcal{G}} \equiv \mathcal{G} \cap (\phi^s \cup \phi^w \cup \phi^r)$ contains the free instantiations in the suggestion made by p_i, and the evidence table is updated as,

$$\begin{aligned} e^{\ell}_{ln} &= P_{new}(C^{\ell}_{lk} = \ell_{\mathcal{G}}) = P(C^{\ell}_{lk} = \ell_{\mathcal{G}} \mid C_{\mathcal{G}} = \ell_{\mathcal{G}})\, P(C_{\mathcal{G}} = \ell_{\mathcal{G}}) \\ &+ P(C^{\ell}_{lk} = \ell_{\mathcal{G}} \mid C_{\mathcal{G}} \neq \ell_{\mathcal{G}})\, [1 - P(C_{\mathcal{G}} = \ell_{\mathcal{G}})] \\ &\approx P(C_{\mathcal{G}} = \ell_{\mathcal{G}}) + P_{old}(C^{\ell}_{lk} = \ell_{\mathcal{G}})[1 - P(C_{\mathcal{G}} = \ell_{\mathcal{G}})] \end{aligned} \tag{5.18}$$

where $\ell_{\mathcal{G}} = \ell_n \in \Omega(y^{\ell})$, $P_{old}(\,\cdot\,)$ is available from E^{ℓ}_{old}, and we set

$$l = \max_j P(C^{\ell}_{jk} = \ell_{\mathcal{G}}), \text{ and } P(C_{\mathcal{G}} = \ell_{\mathcal{G}}) = \begin{cases} 1/|\phi_{\mathcal{G}}|, \text{ if } \ell_{\mathcal{G}} \in \phi_{\mathcal{G}} \\ 0, \text{ if } \ell_{\mathcal{G}} \notin \phi_{\mathcal{G}} \end{cases}.$$

c) *Hard Evidence:* Let $\mathcal{G}$ denote an ICP suggestion that is refuted by p_k (k=1 or 2) who shows a card $\ell_{\mathcal{G}} = \ell_m \in \Omega(y^{\ell})$, and

$$h = \max_j H(C^{\ell}_{jk}), \text{ then } e^{\ell}_{hm} = 1 \text{ and } e^{\ell}_{hl} = 0,$$

for $\forall l \neq m$ in the table E^{ℓ}. Where, H is the information entropy defined in eq. (3.2).

II. The evidence table row and column containing the element e^{ℓ}_{ij} that was updated in Step I are normalized by the following procedure:

- Let $M = |\Omega(y^\ell)|$ be the number of columns in the matrix E^ℓ. Then,

$$\text{if } e^\ell_{il} = 1, \text{ set } e^\ell_{im} = 0, \ \forall\, m \neq l.$$

Otherwise, compute the following variations,

$$\text{let } \Delta e^\ell_i = 1 - \sum_{m=1}^{M} e^\ell_{im}, \text{ then } e^\ell_{il} = e^\ell_{il} + \frac{\Delta e^\ell_i}{M}, \ \forall\, l \neq j, \text{ and } \forall\, e^\ell_{il} \neq 0.$$

Set the elements, e^ℓ_{il}, that are negative equal to zero, and repeat the above procedure until they all are non-negative.

- Consider the column in E^ℓ that contains e^ℓ_{ij}, then

$$\text{if } e^\ell_{kj} = 1, \text{ set } e^\ell_{nj} = 0, \ \forall\, n \neq k, \ n \neq i.$$

Otherwise, leave the column unaltered.

5.3.6 Results: Optimal Game Strategies

The CLUE® decision tree can be obtained by the methodology in Section 4.2, or an influence diagram can be obtained by folding the connectivity tree and augmenting it with observation and test-decision nodes, using the procedure presented in [46], leading to Fig. 5.3. $x(t_k)$ represents the cells that can be visited at time t_k. $z(t_k)$ represents the set of suggestions that can be performed at time t_k. Each action decision, $a(t_k)$, determines the cell to which the pawn moves at time t_{k+1}, and each test decision $u(t_k)$ determines whether one of the admissible suggestions will be performed. Since the turn is lost every time a suggestion is made, the cost of observation $J(t_k)$ is the average distance that can be traveled in one turn. The tradeoff between value and cost of observation is specified through the weights w_B, w_D, and w_J, obtaining an ICP that favors moving quickly about the board versus entering the rooms when $w_B << w_D, w_J$, and viceversa.

The set of CLUE® measurements consists of the suggestions that can be performed inside the nine rooms in the mansion, $M = \{m_d,\, m_l,\, m_b,\, m_h,\, m_k,\, m_g,\, m_a,\, m_c\}$. Each measurement in M can be performed only inside its room, and its outcome is unknown *a priori*. However, since its posterior probability can be computed from $P(y^r, M)$, it is used to estimate the information reward through eq. (5.1). Let the suggestions made in the seven rooms that do not belong to the ICP be denoted by $M^C = \{m_\iota : \iota = 1, \ldots, 7,\ \ r_\iota \in \Omega(y^r)\}$. Then, each suggestions in M^C has four mutually-exclusive outcomes, defined as: r_ι belongs to p_1 (m_ι^1), r_ι does not belong to p_1 but belongs to p_2 (m_ι^2), r_ι does not belong to neither p_1 nor p_2 (m_ι^3), and $r_\iota = un$ (m_ι^4). Each outcome corresponds to a possible reply by p_1, and p_2 (if p_1 has none of the suggested cards), in response to an ICP suggestion in room r_ι. Thus, the CPT of a suggestion $m_\iota \in M^C$ in the CLUE® BN (Fig. 5.8) is,

$$
\begin{aligned}
&P \quad (m_\iota = m_\iota^1 \mid C_{l1}^r = r_\iota,\, C_{ji}^r) = 1, \ \text{ if } \ \upsilon(pa(m_\iota)) = 1, \ \text{ for } \ l = 1, 2, 3 \\
&P \quad (m_\iota = m_\iota^2 \mid C_{l2}^r = r_\iota,\, C_{ji}^r) = 1, \ \text{ if } \ \upsilon(pa(m_\iota)) = 1, \ \text{ for } \ l = 4, 5, 6 \\
&P \quad (m_\iota = m_\iota^3 \mid C_{ji}^r \neq r_\iota) = 1, \ \text{ if } \ \upsilon(pa(m_\iota)) = 1, \ \text{ for } \ \forall j, i \\
&P \quad (m_\iota = m_\iota^4 \mid C_{ji}^r) = 1, \ \text{ if } \ \upsilon(pa(m_\iota)) = 0,
\end{aligned}
\tag{5.19}
$$

and is defined for $\forall C_{ji}^r \in ch(y^r)$, where $\upsilon(\,\cdot\,)$ is given by eq. (5.12), and all other CPT entries are set equal to zero. Then, the CLUE® BN and the evidence tables (Section 5.3.5) are used to evaluate the information benefit, as shown in Section 3.3.

Every time $P(y^r \mid E^r)$ undergoes a substantial change, the CLUE® influence diagram is used to obtain a new optimal plan for the ICP moves, using the exact LIMIDs algorithm [54], implemented by the Bayesian Network Toolbox for Matlab [56]. Some examples are provided in Table 5.2, for various combinations of q_0 and q_f, and of the weights w_B and w_J. It can be seen from these paths that when the cost of observation is weighted highly with respect to its benefit ($w_J >> w_B$) the paths tend to

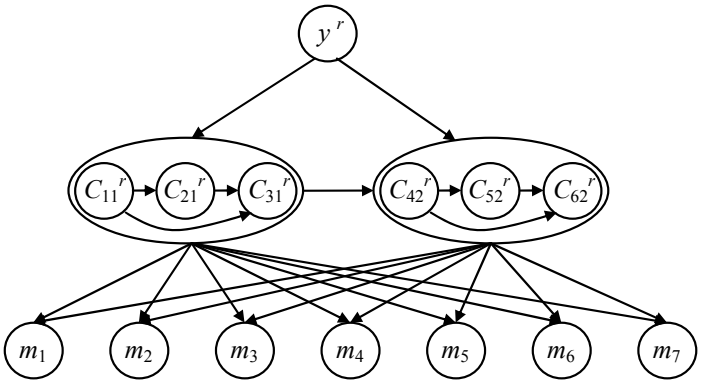


Figure 5.8: BN model of the CLUE® suggestions that may be performed in the seven rooms that are in the domain of the hidden room card, $\Omega(y^r)$.

have fewer observation cells, unless entering a room shortens the distance traveled by means of a secret passage. In order to verify the approach, the paths obtained with w_B set equal to zero are illustrated in Table 5.2. In this case, the test decisions are always negative, and the paths tend to include only void cells. On the other hand, when $w_J << w_B$, the paths include several observation cells and, typically, corresponding positive test decisions. It also is possible to move through an observation cell (e.g., a in the first row of Table 5.2) with a negative test decision, indicating that a suggestion is not made but the cell is visited to minimize distance.

5.3.7 Results: Comparisons to Other Methods

We tested the ICP by making it compete against human players (HP) and the CSP player (Table 5.3) through the interactive simulation described in Section 5.2. In these games, the ICP is fully autonomous. It implements the paths obtained from the ID to move its pawn, and utilizes the CLUE® BN to make suggestions inside the rooms visited along the path. The CSP player implements a constraint satisfaction approach [32, Chapter 5], with two types of constraints: (C1) $y^\ell \neq C^\ell_{ji}$, for $\forall\ \ell, i, j$,

Table 5.2: Optimal CLUE® paths obtained using different benefit and cost weights.

q_0, q_f	w_B, w_J	**Optimal cell sequence**	$B(T)$	$J(T)$	$D(T)$
1, 18	0, 0	[1 2 3 7 a 22 20 17 16 18]	0	0	10.5
	0, 12	[1 2 3 4 5 12 13 16 18]	0	0	11.5
	12, 0	[1 2 3 4 5 6 c 6 5 7 a 22 20 17 16 18]	9.55	14.5	0
1, 51	0, 0	[1 2 3 4 5 6 c g 64 63 62 d 51]	0	0	10.5
	0, 6	[1 2 3 7 14 15 17 19 21 23 25 27 52 51]	0	0	21.5
	10, 6	[1 2 3 4 5 6 c g 64 63 62 d 51]	7.96	12	10.5
100, 22	0, 0	[100 97 s k 46 39 38 37 40 a 22]	0	0	7
	0, 6	[100 97 94 91 92 89 87 85 83 79 76 19 17 20 22]	0	0	18
	12, 0	[100 97 98 99 h 99 98 97 s k 46 39 38 37 40 a 22]	13.84	0	11

and (C2) $C^{\ell}_{ji} \neq C^{\ell}_{nm}$, for $i, m = 2, 3$, $j, n = 1, 2, 3$, and $i \neq m$ or $j \neq n$. When the CSP makes a suggestion G, and the i^{th} player refutes it by showing its own card ℓ_G, the following value is obtained $C^{\ell}_{ji} = \ell_G$, for some j. In order to make a suggestion, the CSP searches an assignment for y^s, y^w, and y^r that satisfies (C1)-(C2). Based on a heuristic strategy recommended by the CLUE® game rules, the CSP pawn moves to the nearest room to begin making suggestions. Then, it moves back and forth between two adjacent rooms to determine y^s, y^w, and subsequently navigates the board to determine y^r. The CSP and a naive version of the ICP both use a heuristic rule developed in [31] to navigate the board without implementing the ID developed in this dissertation.

Table 5.3: Game results for the ICP competing against HP and CSP

Winning Player:	**ICP**	**HP**	**CSP**
ICP with ID	48.0%	40.0%	12.0%
ICP without ID	36.0%	44.0%	20.0%

The game results in Table 5.3 show that, by utilizing the ID, the ICP wins more often than the HP or the CSP. Without ID, an ICP implementing only the CLUE® BN (proposed in [31]) wins more often than the CSP, but less often than the HP.

Recently, a CLUE® player implementing a neural network and Q-learning was presented in [12]. Its framework is illustrated in Fig. 5.9. The basic idea to develop the neural player is to train a neural network to approximate the decision-value function representing the value of information, for which there exists no general closed-form representation (the incremental entropy is not assured to be the best value function). Bayesian inference, test (suggestions), and action (motion) decision making are unified using an MDP framework. For more details, refer to [12]. Its winning rates show that while it beats the BN and CSP player 60% and 75% of the times, respectively, it performs similarly to the HP. Therefore, based on Tables 5.3 and 5.4, the ICP with ID can be considered as the most effective of all of the aforementioned players. The game records show that, typically, the player who first infers the hidden room card y^r correctly wins the game, and that the ICP typically does so before the other players. Thus, its success can indeed be attributed to its strategy for selecting the optimal pawn movements and suggestions that maximize the information reward (2.2) associated with y^r.

Table 5.4: Game results for the Neural Player competing against CSP and ICP without ID

Winning Player:	**Neural Player**	**CSP**
Winning Rate	75.0%	25.0%
Winning Player:	**Neural Player**	**ICP without ID**
Winning Rate	60.0%	40.0%

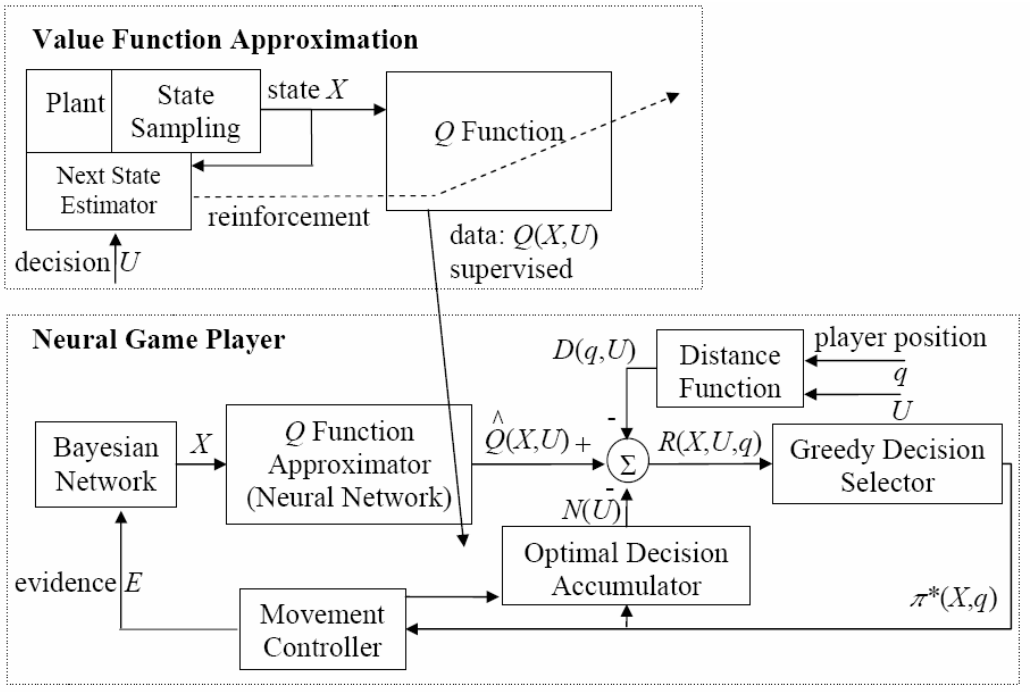


Figure 5.9: MDP Neural Player; Bayesian inference, test (suggestions), and action (motion) decision making are unified using an MDP framework.

Chapter 6

Application II: Feature-level Fusion and Target Classification in Robotic Demining

The treasure hunt problem and the robotic sensor planning methodology presented in Section 2, 3 and 4 is also demonstrated through a demining application. The simulated minefield (Section 6.1) contains multiple obstacles and targets of various size and geometry. The obstacles may consist of buildings, bodies of water, or trees that must be avoided by the robot platform. The targets consist of buried mines and clutter objects characterized by various features, such as, metal content, depth, and shape, that can be used to classify them as either mines or false alarms. Since the targets are buried, the features must be extracted from raw measurements, and their accuracy depends on the environmental conditions and on the sensor type and mode by which the measurements are obtained.

Typically, multiple and heterogeneous sensor measurements are required to classify the targets with reasonable accuracy [35]. We consider a scenario that commonly arises in modern demining systems [57]. An infrared (IR) sensor is initially deployed on an airborne platform, such as, an autonomous air vehicle (UAV), to obtain cursory measurements from the entire minefield. Subsequently, a more accurate ground penetrating radar (GPR) sensor is deployed on a ground robot platform, which may be overpass capable, to obtain additional measurements. Using feature-level fusion [30, 35], the GPR and IR measurements can be combined to improve the accuracy of target classification, such that additional resources can be deployed to excavate the mines.

Since the deployment of more capable sensors, such as GPR installed on ground

robots, can be very costly, it is desirable to plan the sensor motions and measurements such that their effectiveness is maximized and the costs minimized. Thus, the methodology presented in Sections 4 is implemented to obtain an optimal policy σ^*, including the path τ^* and measurement sequence Z_T^*, for a GPR sensor installed on a ground robot. The optimal policy takes into account prior information that consist of the IR sensor mode and measurements, and of the environmental conditions in the minefield. Although we assume the environmental conditions to be the same for the IR and GPR sensor, the method is also applicable if the environment changes after the IR measurements are obtained. The geometry and location of obstacles and potential targets are obtained from prior airborne IR sensors, cameras, and maps of the minefield of interest.

6.1 Demining System Simulation

The sensor planning methodology is tested on the simulation of a landmine sensing system developed in [35], which includes sensors, targets, obstacles, soils, and meteorological conditions. A grid is superimposed on a rectangular minefield dividing it into square *bins*, where the grid spacing is equal to one unit distance. Soil composition (e.g., clay or sand), soil characteristics (e.g., magnetic properties, moisture, uniformity), vegetation, and time-varying meteorological conditions are modeled according to [36, 57–59], and can be assigned to each bin in the minefield at random or at user-specified positions. The environmental conditions are assumed uniform inside each bin. As shown in [35], GPR and IR measurements of the target features are reproduced and deteriorated according to the operating conditions and surrounding environment.

Anti-tank mines (ATM), anti-personnel mines (APM), unexploded ordnance (UXO), and clutter objects (CLUT) are reproduced based on the Ordata Database [36] and

placed in the field with a pre-determined geometry. Every target has associated actual features of depth d, size z, shape s, and metal content c (Table 6.1). When a sensor detects an object, the simulation uses the actual target features and environmental/meteorological conditions to generate sensor data with random noise and errors that are commensurate to the given situation (see [60] for more details). Subsequently, a set of measured features that depends on the sensor and the environmental conditions is obtained for the target. The sensor platform is simulated by a rectangle or triangle with sensor field of view denoted by a triangle geometry $\mathcal{S}$ mounted on the platform. When the sensor is turned on, the senor field of view appears along the robot motion path and the simulation produces measured features for the region intersected by its field-of-view, based on its geometry $\mathcal{S}$ and configuration q in the minefield.

6.2 IR and GPR Sensors and Bayesian Network Models

Infrared sensors detect anomalies in infrared radiation that is either emitted by mines, soil, or vegetation. Based on the location of the sensor, the radiation data can be processed to build an image of an horizontal area and to estimate the depth of the object therein. Images can be obtained for depths up to 12 cm. The mode of IR sensors, m_{IR}, influences the measured target features and is uniquely determined by its height above the ground. Therefore, airborne IR sensors typically obtain only cursory measurements of size and shape for shallow-buried objects. Because they rely on temperature variations, their performance also is highly influenced by environmental conditions, such as, time, weather, vegetation, and soil properties. Using the methodology reviewed in Section 3.1 and presented in [35], a BN model is obtained for an Agema Thermovision 900 sensor [36, 57–59]. The BN architecture

and node definitions are shown in Figure 6.1 and Table 6.1, respectively.

GPR sensors emit radio waves that penetrate the ground and process their reflections at the boundaries of materials characterized by different refraction indexes. Images of underground vertical slices and of any objects buried within is obtained over $\mathcal{S}$ by sensing discontinuities in electrical properties. The measured size, shape, and depth of an underground object can be obtained from these images through signal processing techniques [61], such as edge extraction. The frequency of the radio wave and its bandwidth determine the search mode m_{GPR}. Since penetration depth increases at lower frequencies and image resolution improves at higher frequencies, the optimal GPR mode depends on target features and on environmental conditions. For example, very high frequencies may be required in the presence of ground discontinuities to overcome the so-called ground-bounce effect (GBE) [57]. The GPR BN model illustrated in Fig. 6.1, taken from [35], is learned from data and prior expert knowledge. After the IR and GPR models are learned, given IR or GPR measurements, the posterior distribution of the feature set $\{d, z, s\}$ can be inferred. Then the posterior distribution of the feature set $\{d, z, s\}$ is viewed as soft evidence and is entered to the BN classifier shown in Fig. 6.2 to infer the posterior distribution of hypothesis variable y representing if the target is a mine or clutter.

Using the methodology developed in Chapter 4 and [30], the relationships between sensor mode, environmental conditions, and feature-level Dempster-Shafer fusion performance are accounted for in planning the optimal GPR path and measurement sequence, given prior IR measurements.

6.3 Optimal GPR Sensor Planning

The GPR sensor is installed on-board a robotic platform with geometry $\mathcal{A}$, and field-of-view $\mathcal{S}$. After the minefield, or $\mathcal{W}$, is simulated, the cell decomposition $\mathcal{K}$

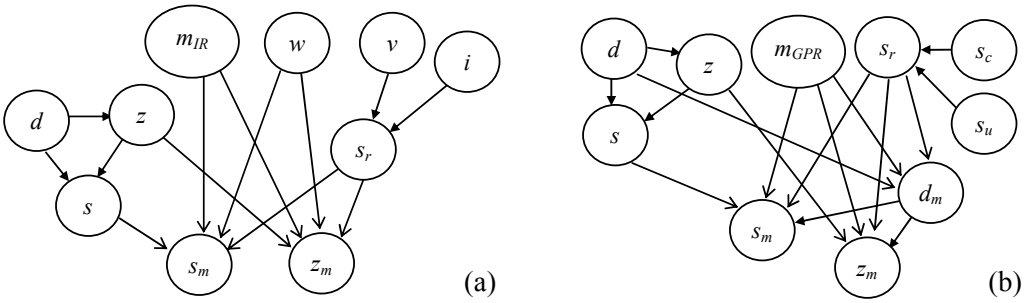


Figure 6.1: Architectures of IR and GPR BN sensor models (taken from [35]), with nodes defined in Table 6.1.

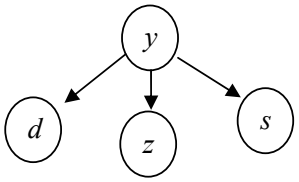


Figure 6.2: Architectures of BN Classifier (taken from [60]), with nodes defined in Table 6.1 and hypothesis variable y.

is obtained using the methodology in Section 4.1. Then a connectivity graph with observations, $\mathcal{G}$ is constructed from $\mathcal{K}$. A decision tree DT, obtained by the transformations $\mathcal{G} \Rightarrow T_r \Rightarrow DT$, is utilized to compute the optimal policy σ^* of path planning and sensor measurement planning for the GPR sensor platform.

The optimal channel τ^* along the optimal policy σ^* is transformed into a free path for the robot $\mathcal{A}$ by connecting the centroid of initial cell κ_0 and the centroid of final cell κ_f through the midpoints of the intersections of every two successive cells in τ^*.

For comparison to the optimal path τ^*, four different methods, A* algorithm, complete coverage, random coverage and fixed grid search, are implemented to plan the robot's path. The shortest path τ_{short} is searched from the connectivity graph $\mathcal{G}$ by A* algorithm [48] given initial configuration $q_0 \in \kappa_0$ and final configuration $q_f \in \kappa_f$,

Table 6.1: List of nodes in Bayesian network models of GPR and IR sensors

Set:	Node:	Instantiations and corresponding range:
V	GPR mode: m_{GPR}	depth search, resolution search, anti ground-bounce-effect search
	IR mode: m_{IR}	surface-mine search, shallow-buried-mine search
E	Soil moisture (%): s_r	dry [0-10], wet (10-40], saturated (> 40)
	Soil composition: s_c	very-sandy, sandy, high-clay, clay, silt
	Soil uniformity: s_u	yes, no
	Vegetation: v	no-vegetation, sparse, dense
	Weather: w	clear, overcast, raining
	Illumination: i	low (7-10 a.m. and 6-9 p.m.), medium (10-1 p.m.), high (1-6 p.m.)
F	Depth (cm): d	surface [0], shallow-buried (0-12], buried (12-60], deep-buried (> 60)
	Size (cm): z	small (2-13], medium (13-24], large (24-40], extra-large (> 40)
	Shape: s	cylinder, box, sphere, long-slender, irregular

based on the the same distance definition as in τ^*, including rotation distance. The measurements along τ_{short} are taken using a fixed GPR sensor mode of depth search and knowledge of the environmental conditions is only used as evidence in processing GPR sensor measurements for inference, when the path is being executed.

The method of complete coverage motivated from [2] and [62] generates a path, τ_{cover}, along which all the targets in $\mathcal{W}$ are able to be observed by the sensor. In a workspace without obstacle, complete coverage performs simple back-and-forth motions. In a workspace $\mathcal{W}$ with obstacles, the motion keynote of complete coverage is still back and forth, but the robot has to rotate around the obstacles and go through the areas unable to be covered the back and forth motions. Let $\mathcal{W}_{covered} \subseteq \mathcal{W}$ denote the workspace where all the targets in $\mathcal{W}_{covered}$ are able to be observed by the sensor along path τ_{cover}. $\mathcal{W}_{uncovered} = \mathcal{W} \backslash \mathcal{W}_{covered}$. Let $\mathcal{L}_{cover}$ denote the set of observation cells τ_{cover} goes though in order from the initial configuration q_0. Based on the

connectivity graph $\mathcal{G}$ obtained by the cell decomposition presented in section 4.1, the path of τ_{cover} is generated using the following algorithm: (1) choose a configuration q_0 in the upper right corner of $\mathcal{W}$ so that $q_0 \in$ an observation cell $\bar{\kappa}_0$, then add $\bar{\kappa}_0$ into the list of $\mathcal{L}_{cover}$ and set τ_{cover} initially be empty; (2) in the back-forth direction, every approximate h bins distance, choose a configuration $q_i \in \bar{\kappa}_i \subseteq \mathcal{W}_{uncovered}$, then add $\bar{\kappa}_i$ into $\mathcal{L}_{cover}$; (3) Using A*, search a shortest path τ from $\bar{\kappa}_{i-1}$ to $\bar{\kappa}_i$ through $\mathcal{G}$, combine τ and the old τ_{cover} into a new path from $\bar{\kappa}_0$ to $\bar{\kappa}_i$, and replace the old τ_{cover} by setting the new path to be τ_{cover}; (4) repeat (2) and (3) from $i = 1$ until the back-forth motions have to stop; (5) If there is an connected area $\mathcal{W}_{connected} \subseteq \mathcal{W}_{uncovered}$, perform (1)-(4) to generate a path $\tau_{cover_{sub}}$ and a new list $\mathcal{L}_{cover_{sub}}$; (6) find the two closest cells κ_i and κ_j in τ_{cover} to $\tau_{cover_{sub}}$; (7) break τ_{cover} at the cells κ_i and κ_j, insert $\tau_{cover_{sub}}$ to form a new path, set it to be the new τ_{cover}, according add the members in $\mathcal{L}_{cover_{sub}}$ into $\mathcal{L}_{cover}$ by the order from $q_0 \in \bar{\kappa}_0$; (8) repeat (5)-(7) until $\mathcal{W}_{covered} = \mathcal{W}$, and output τ_{cover} and the list of observation cells $\mathcal{L}_{cover}$.

Random coverage [2] [62], τ_{rand}, is generated with the following algorithm: (1) select the initial position $q_0 \in \bar{\kappa}_0 \in \mathcal{L}_{cover}$; (2) randomly select the second cell $\bar{\kappa}_i$ from $\mathcal{L}_{cover} \backslash \{\bar{\kappa}_0, \ldots, \bar{\kappa}_{i-1}\}, i = 1, \ldots, 0.5|\mathcal{L}_{cover}|$, where $|\cdot|$ denotes the cardinality of the set of $\mathcal{L}_{cover}$; (3) using A* search a path τ_i from $\bar{\kappa}_{i-1}$ to $\bar{\kappa}_i$; (4) combine all the paths in series $\tau_i, i = 1, \ldots, 0.5|\mathcal{L}_{cover}|$ to obtain a random coverage path τ_{rand}.

The fixed grid path, τ_{grid}, which is used in [17] and in a different version in [2], is generated using following algorithm: (1) Given initial configuration $q_0 \in \kappa_0$ and final configuration $q_f \in \kappa_f$, find the closest observation cell $\bar{\kappa}_i \in \mathcal{L}_{cover}$ to κ_0 and the closest cell $\bar{\kappa}_j \in \mathcal{L}_{cover}$ to κ_f; (2) Get the sequence $\bar{\kappa}_i, \ldots, \bar{\kappa}_j$ directly out of $\mathcal{L}_{cover}$ and generate another sequence $\kappa_0, \bar{\kappa}_i, \ldots, \bar{\kappa}_j, \kappa_f$; (3) view members in the new sequence as fixed grids, use A* algorithm to obtain the shortest path between each pair of the adjacent grids and combine these paths in series to generate τ_{grid}. The GPR mode

along τ_{grid} is fixed.

6.4 Performance Metrics

One of the main challenges of sensor planning is that the sensor path must be planned before the targets are properly classified. So although the observation benefit, (3.20), reflects the value of a measurement sequence, the actual classification improvement cannot be assessed *a priori*. Therefore, the methodology presented in the previous section is evaluated by computing efficiency metrics after the sensor policy has been executed, and GPR measurements have been obtained from all the targets along the sensor path. Let ΔN_y denote the number of targets properly classified after fusing GPR and IR measurements minus the targets properly classified before deploying the GPR, based solely on IR measurements. Then, the efficiency metric $\eta_y(\sigma) = \Delta N_y(\sigma)/D(\tau)$ is the classification improvement per unit distance of a policy σ, inducing the robot path τ. Let $N_{\mathcal{T}}$ denote the number of observed targets along path τ. Then, the efficiency metric $\eta_{\mathcal{T}}(\sigma) = N_{\mathcal{T}}(\sigma)/D(\tau)$ is the number of observed targets per unit distance along τ.

As shown in [35], one advantage of using BN models to process sensor measurements and infer the target features and classification is that a confidence level (CL) is provided in the form of a posterior probability $P(F_i, y_i \mid e)$, given the evidence for a target $\mathcal{T}_i$. Thus, after both IR and GPR measurements are obtained, the state of y_i can be estimated along with a CL representing the probability that the estimate provided is correct. An error metric that accounts for CL is obtained by weighting the distance from the true value of the inferred variable by the respective probability:

$$J_y(\mathcal{T}_i|e) = \mathbf{p}_{y_i} \cdot \mathbf{g}_{y_i} \tag{6.1}$$

where $\mathbf{p}_{y_i}$ is a $1 \times p$ vector of probabilities obtained by the inference algorithm:

$\mathbf{p}_{y_i} \equiv [P(y_i^1 \mid e) \cdots P(y_i^p \mid e)]$. $\mathbf{g}_{y_i}$ is a $p \times 1$ vector containing the distance g_{y_i} between the true value of y_i (i.e., y_i^v) and all of its possible instantiations: $\mathbf{g}_{y_i} \equiv [g_{y_i}(y_i^1, y_i^v) \cdots g_{y_i}(y_i^p, y_i^v)]$. The distance g_{y_i} is a discrete metric that is feature specific and satisfies the following conditions: $J_y(\mathcal{T}_i) = 0.5$, when $\mathbf{p}_{y_i}$ is uniform, and at the worst estimation, $J_y(\mathcal{T}_i) = 1$ is the maximum value of eq. 6.1 [60] .

Let $J_y(\mathcal{T}_i|e^{IR})$ denote the error metric before deploying the GPR, based solely on IR measurements for $\mathcal{T}_i$ and $J_y(\mathcal{T}_i|e^{IR}, e^{GPR})$ denote the error metric after fusing GPR and IR measurements. Let $\Delta J_y(\mathcal{T}_i) = J_y(\mathcal{T}_i|e^{IR}) - J_y(\mathcal{T}_i|e^{IR}, e^{GPR})$. Therefore, ΔJ_y is the error reduction brought by the GPR measurement. Let $\Delta J_y^\sigma = \sum_{\mathcal{T}_i \in T_\tau} \Delta J_y(\mathcal{T}_i)$ denote the total error reduction over a policy σ, where T_τ denote the set of observed targets given a policy σ corresponding to the measurement sequence Z_T. Then, the efficiency metric $\eta_{J_y}(\sigma) = \Delta J_y^\sigma / D(\tau)$ is the error reduction per unit distance of a policy σ, inducing τ.

Let $\Delta H(\mathcal{T}_i) = H(y_i \mid \mathcal{M}_i^{IR}) - H(y_i \mid \mathcal{M}_i^{GPR}, \mathcal{M}_i^{IR})$ denote the actual (not expected) entropy reduction brought by taking GPR measurements $\mathcal{M}_i^{GPR}$ over target $\mathcal{T}_i$. Let $\Delta H_\sigma = \sum_{\mathcal{T}_i \in T_\tau} \Delta H(\mathcal{T}_i)$ denote the total entropy reduction given the policy σ. Then, the efficiency metric $\eta_H(\sigma) = \Delta H_\tau / D(\tau)$ is the entropy reduction per unit distance of a policy σ, inducing τ.

Chapter 7

Application II: Robotic Demining Results

7.1 Influence of Measurements and Environmental Information on Sensor Path

7.1.1 Influence of Target Presence

Fig. 7.1 is used to illustrate that the presence of targets and obstacles must be accounted for in planning the motions of the robot. Suppose there are three equally important targets in the workspace, as shown in Fig. 7.1. If the robot travels to the left of the obstacle, it will necessarily miss the two targets located to the right of it. The path illustrated in Fig. 7.1 is the optimal path computed by our methodology.

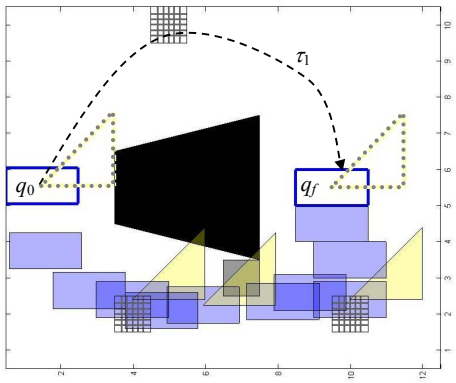
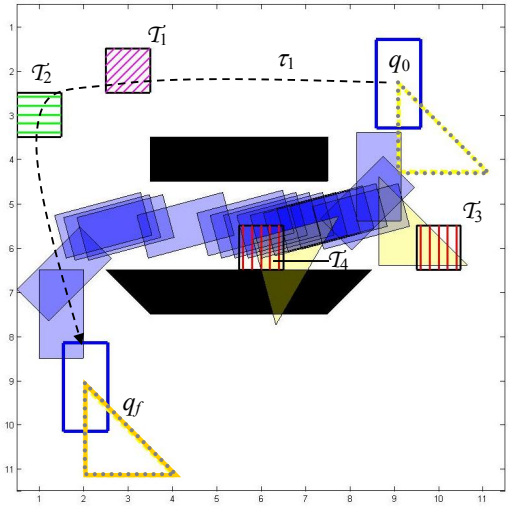


Figure 7.1: Example: minefield with 4 potential targets. Environment conditions are constant through this minefield. Two candidate paths τ^* and τ_1 from initial position q_0 to final position q_f in the minefield. The Robot geometry is denoted by $\mathcal{A}$; the sensor field of view is denoted by $\mathcal{S}$.

7.1.2 Influence of Prior Sensor Measurements

Fig. 7.2 is used to illustrate that prior IR sensor measurements must be accounted for in planning the motions of the robot. Suppose there are two candidate paths of approximately equal distance and approximately equal number of targets, as shown in Fig. 7.2. The environment conditions are constant throughout the minefield and the GPR mode is optimally selected. Based on prior IR measurements, the targets' information benefit, i.e., EER, can be computed by eq. 3.20. The robot travels through the obstacles and takes measurements of the two important targets located between the obstacles. The path illustrated in Fig. 7.2 is the optimal path computed by our methodology.

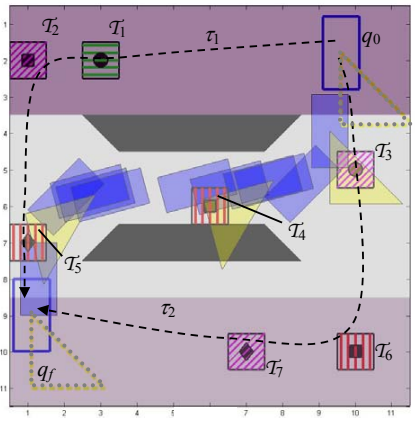


Performance		
Path	τ^*	τ_1
Targets	$\mathcal{T}_3$, $\mathcal{T}_4$	$\mathcal{T}_1$, $\mathcal{T}_2$
$D(t_f)$	20.33	18.91
$B(t_f)$	14.62	3.85
η_y	0.0970	0

Figure 7.2: Example: minefield with 4 potential targets. Environment conditions are constant through this minefield. Red: the highest information benefit (Target 3 and 4); Magenta: intermediate information benefit (Target 1); Green: low information benefit (Target 2). Highest information benefit means EER > 0.2; low information benefit means EER < 0.1; intermediate information benefit means EER is between 0.1 and 0.2. Two candidate paths τ^* and τ_1 from initial position q_0 to final position q_f in the minefield.

7.1.3 Influence of Environmental Conditions

Fig. 7.3 is used to illustrate that influence of environmental conditions must be accounted for in planning the motions of the robot. Suppose there are multiple paths of approximately equal distance and with approximately the same number of targets, as shown in Fig. 7.3. Our methods chooses the one along which the environmental conditions are the most favorable to GPR measurements.



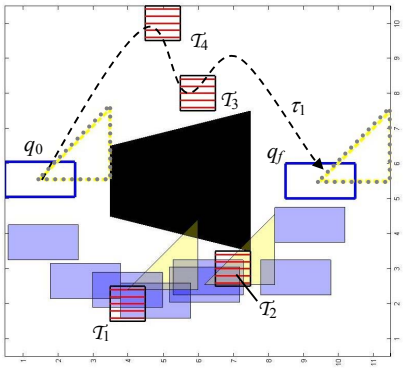
Performance			
Path	τ^*	τ_1	τ_2
Targets	$\mathcal{T}_3$, $\mathcal{T}_4$, $\mathcal{T}_5$	$\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_5$	$\mathcal{T}_3$, $\mathcal{T}_6$, $\mathcal{T}_7$
$D(t_f)$	19.46	18.78	19.32
$B(t_f)$	16.75	9.66	10.06
η_y	0.1542	0.0532	0.1035

Soil: Dry Wet Saturated

Figure 7.3: Example: minefield with 7 potential targets. Red: the highest information benefit (Target 4, 5 and 6); Magenta: intermediate information benefit (Target 2, 3 and 7); Green: low information benefit (Target 1). Target 1 and target 3 are identical but buried under different environmental conditions; Target 2, 4 and 6 are identical but buried under different environmental conditions; Target 5 and 7 are identical but buried under different environmental conditions. Three candidate paths τ^*, and τ_1 and τ_2 from initial position q_0 to final position q_f in the minefield.

7.1.4 Influence of Sensor Mode

Fig. 7.4 is used to illustrate that influence of sensor mode must be accounted for in planning the motions of the robot. Suppose that if the GPR mode is fixed a priori, then the candidate two paths would have approximately the same total reward, but when the GPR mode is optimized, one path has higher reward then the other, as shown in Fig. 7.4. Our method chooses the one along which leads to the best classification improvement when the sensor mode is optimized.



Performance		
Path	τ^*	τ_1
Targets	$\mathcal{T}_1$, $\mathcal{T}_2$	$\mathcal{T}_3$, $\mathcal{T}_4$
$D(t_f)$	12.78	12.04
Fixed GPR mode, η_{J_y}	0.0250	0.0191
Optimal GPR mode, $\eta_{J_y}^*$	0.0382	0.0143

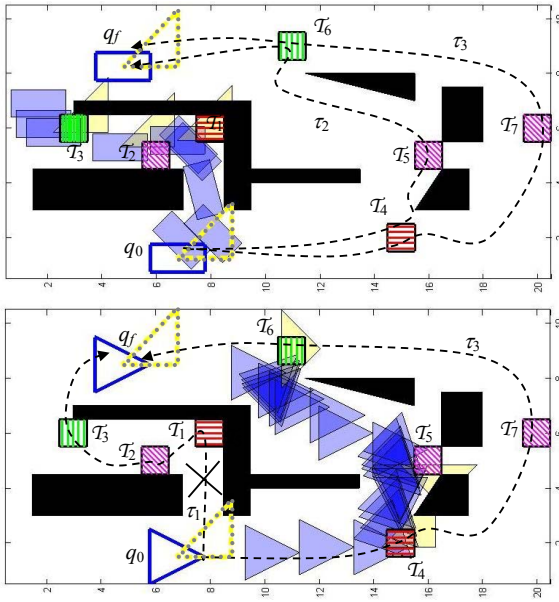
Figure 7.4: Example: minefield with 4 potential targets. Two candidate paths τ^* and τ_1 from initial position q_0 to final position q_f in the minefield.

7.1.5 Influence of Robot Geometry on the Optimal Path

Fig. 7.5 is used to illustrate that influence of robot geometry must be accounted for in planning the motions of the robot. Suppose there are several candidate paths of approximately equal information benefit, as shown in Fig. 7.5. By taking account robot geometry, the path with overall shortest distance (including rotation distance) illustrated in Fig. 7.5 is computed by our methodology to achieve the highest path efficiency. Two robots $\mathcal{A}_1$ and $\mathcal{A}_2$ of different geometry carrying the same GPR sensor are deployed in the minefield shown in Fig. 7.5.

7.1.6 Influence of Sensor Geometry on the Optimal Path

Fig. 7.6 is used to illustrate that influence of sensor geometry must be accounted for in planning the motions of the robot. Suppose that because of the geometry of the field of view , once the robot gets sufficiently near a target, it can rotate to take a measurement of the target, as shown in Fig. 7.6. While having longer distance, the optimal path illustrated in Fig. 7.6 is computed by our methodology so that it has a higher information benefit due to the high EER target along the way.



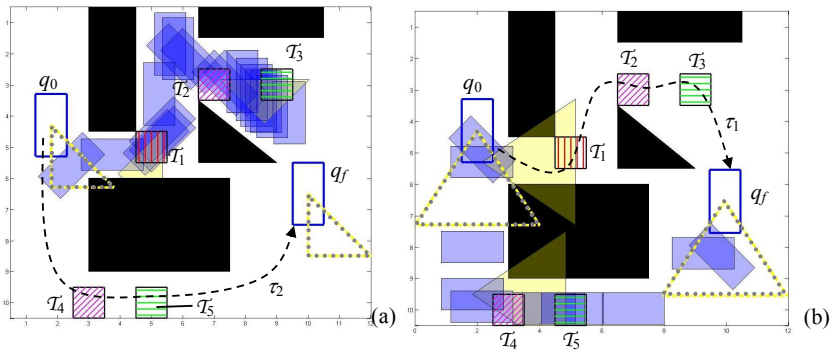
(a)

Robot geometry $\mathbf{A}_1$			
Path	$\tau^* = \tau_1$	τ_2	τ_3
Targets	$\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$	$\mathcal{T}_4$, $\mathcal{T}_5$, $\mathcal{T}_6$	$\mathcal{T}_4$, $\mathcal{T}_6$, $\mathcal{T}_7$
$D(t_f)$	20.47	27.50	32.95
$B(t_f)$	9.93	9.71	9.91

(b)

Robot geometry $\mathbf{A}_2$		
Path	$\tau^* = \tau_2$	τ_3
Targets	$\mathcal{T}_4$, $\mathcal{T}_5$, $\mathcal{T}_6$	$\mathcal{T}_4$, $\mathcal{T}_6$, $\mathcal{T}_7$
$D(t_f)$	30.94	33.59
$B(t_f)$	9.71	9.91

Figure 7.5: Example: minefield with 7 different potential targets. Environment conditions are different through this minefield. Red: the highest information benefit (Target 1 and 4); Magenta: intermediate information benefit (Target 2, 5 and 7); Green: low information benefit (Target 3 and 6). Three candidate paths for robot $\mathcal{A}_1$ and two candidate paths for robot $\mathcal{A}_2$.



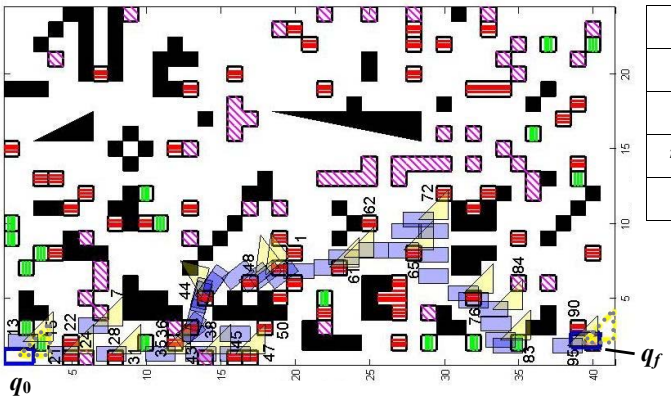
Sensor geometry $\mathbf{S_1}$				
Path	Targets	$D(t_f)$	$B(t_f)$	η_{J_y}
τ^*	$\mathcal{T}_1,\mathcal{T}_2,\mathcal{T}_3$	22.6	14.6	0.0381
τ_2	$\mathcal{T}_4,\mathcal{T}_5$	15.1	4.67	0.0226

Sensor geometry $\mathbf{S_2}$				
Path	Targets	$D(t_f)$	$B(t_f)$	η_{J_y}
τ^*	$\mathcal{T}_1,\mathcal{T}_4,\mathcal{T}_5$	21.3	14.4	0.0516
τ_1	$\mathcal{T}_1,\mathcal{T}_2,\mathcal{T}_3$	22.6	14.6	0.0381

Figure 7.6: Example: minefield with 5 different potential targets. Environment conditions are different through this minefield. Red: the highest information benefit (Target 1); Magenta: intermediate information benefit (Target 2 and 4); Green: low information benefit (Target 3 and 5). Two candidate paths for robot $\mathcal{A}$ with sensor field of view $\mathcal{S}_1$.

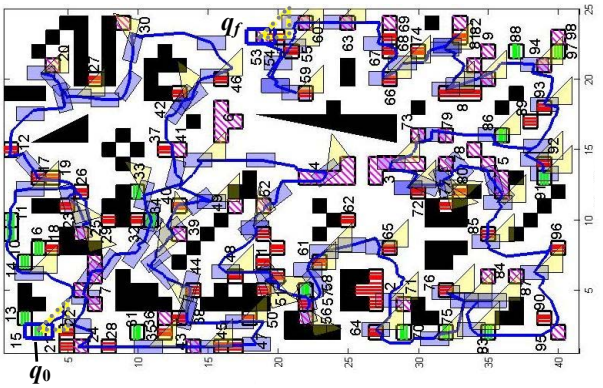
7.2 Path Efficiency in Full Scale Simulations and Comparison with Existing Methods

In this section, the method is demonstrated by computing a robotic sensor path and measurement sequence in a simulated minefield. The path efficiency obtained by our method is compared to the shortest path, complete coverage and random coverage applied to the same minefield, using the same prior information. The comparison shows our method achieves better efficiency by using the metrics $\eta_{\mathcal{T}}$, η_y, η_{J_y} and η_H in Section 6.4. The optimal path τ^* is shown in Fig. 7.7. A complete coverage path is shown in Fig. 7.8.



τ^* efficiency	
η_τ	0.4610
η_y	0.0595
η_{J_y}	0.0446
η_H	0.0599

Figure 7.7: Example: optimal path τ^* from the upper left corner to down left corner in the field ($w_B = 20$, $w_J = 1$) It is a long path in the field and covers 27/98 targets in the field. The robot translates and rotates to pass narrow passages, and tends to take measurements over important targets (colored red and magenta) along the path.
.



τ_{cov} efficiency	
η_τ	0.2683
η_y	0.0055
η_{J_y}	0.0153
η_H	0.0410

Figure 7.8: Example: complete coverage path τ_{cover}. It covers 98/98 targets in the field. A sample of robot/sensor configuration is illustrated along the path. The trajectory of the c.g. is shown in a blue solid line.

Table 7.1: Method efficiency and comparison with other approaches

Average Path Efficiency	Method				
	Optimal	Shortest (Improvement of τ^*)	Complete Coverage (Improvement of τ^*)	Random Coverage (Improvement of τ^*)	Fixed Grid (Improvement of τ^*)
$\eta_{\mathcal{T}} = \frac{N_{\mathcal{T}}}{D}$	0.4610	0.3053 (51.0%)	0.2683 (71.8%)	0.1441 (219.9%)	0.2321 (98.6%)
$\eta_y = \frac{\Delta N_y}{D}$	0.0595	0.0407 (46.2%)	0.0055 (981.8%)	0.0114 (421.9%)	0.0122 (387.7%)
$\eta_{J_y} = \frac{\Delta J_y}{D}$	0.0446	0.0157 (184.1%)	0.0153 (191.5%)	0.0098 (355.1%)	0.0133 (235.3%)
$\eta_H = \frac{\Delta H_\tau}{D}$	0.0599	0.0330 (81.5%)	0.0410 (46.1%)	0.0244 (145.5%)	0.0343 (74.6%)

7.3 Overall Method Efficiency Comparisons

In this section, a representative average for our method is computed for different types of minefields and compared to compare to other methods, such as, complete coverage and random coverage. The average of our method is obtained by averaging the efficiency for several optimal paths planned with different initial and final conditions. Different initial and final configurations will specify different paths computed by our method. Different minefield will yield different path efficiency even using the same method. Therefore, this section is designed to show that our proposed method generally achieves better path efficiency than other methods, not depending on specific initial and final conditions. It is also shown that our method can be used to cover most of the configuration space using several optimal paths and outperforms complete coverage and random coverage methods.

7.3.1 Obstacles Density and Narrow Passages

The example shown in Fig. 7.9 computes paths and efficiency metrics for fields (or regions of the same field) that have different concentration of obstacles and narrow passages. In easy condition, i.e., low obstacles density case, given a fixed cost, our method tends to cover more targets than in harsh condition, i.e., high obstacles density case. Therefore, the performance of our method depends on obstacle geometries of different density. Based on the average classification gain shown in Fig. 7.10 and similar results of other path efficiency metrics, such as, error reduction, the proposed method shows a better efficiency than complete coverage and random coverage methods under obstacle geometries of different density.

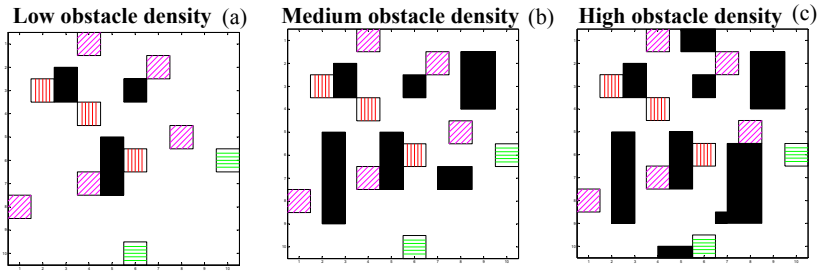


Figure 7.9: Example: minefields of different obstacle density.

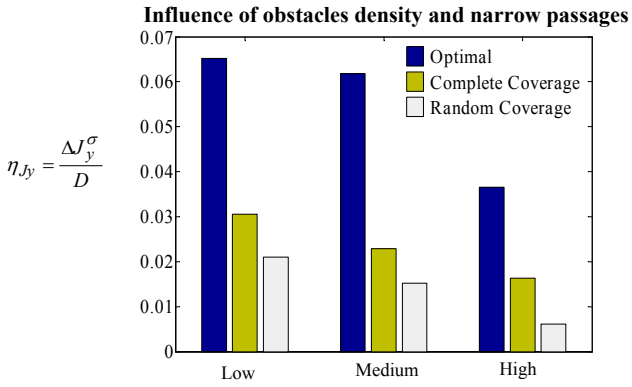


Figure 7.10: The average classification gain $\eta_{J_y} = \Delta J_y^\sigma / D(\tau)$ for the three minefields shown in Fig 7.9.

7.3.2 Target Density

The example shown in Fig. 7.11 computes paths and efficiency metrics for fields (or regions of the same field) that have different concentration of targets and the same density of obstacles. Based on the average classification gain shown in Fig. 7.12 and similar results of other path efficiency metrics, the proposed method shows a better efficiency than complete coverage and random coverage methods under target geometries of different density. The most significant improvement is obtained for high target density case, because in this case, our method easily covers more targets in a given cost than the other two density cases.

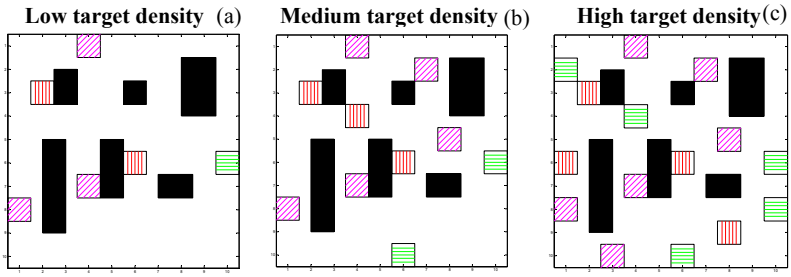


Figure 7.11: Example: minefields of different target density.

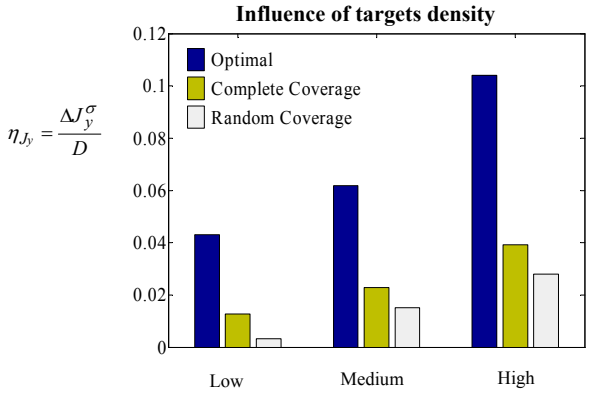


Figure 7.12: The average classification gain $\eta_{J_y} = \Delta J_y^\sigma / D(\tau)$ for the three minefields shown in Fig. 7.11.

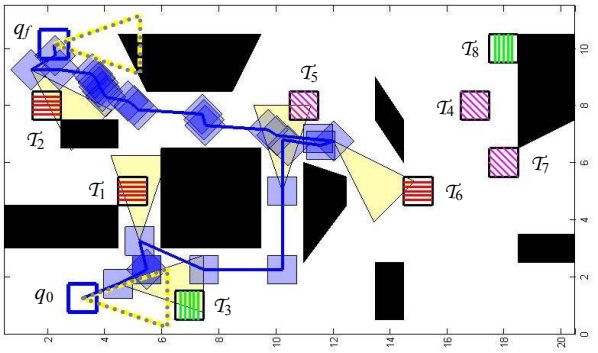
7.4 Non-overpass Capable Platforms

This is important in the case of a non-overpass capable platform, which may be destroyed by driving over a landmine. In the example shown in Fig. 7.13, the robot has a square geometry and the sensor field of view is an isoceles triangle. The optimal path, illustrated in Fig. 7.13, shows that all of the following objectives are achieved,

(1) the robot avoids robot collisions with obstacles and mines;

(2) the sensor is able to take measurements over targets;

(3) the optimal path displays good path efficiency, that is comparable to the average path efficiency displayed in mild target density (Section 7.3.2).



τ^* efficiency	
η_τ	0.1575
η_y	0.0630
η_{J_y}	0.0451
Targets	$\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$, $\mathcal{T}_5$, $\mathcal{T}_6$

Figure 7.13: Non-overpass capable robot example: minefield with 8 different potential targets . Environment conditions are different through this minefield. Red: the highest information benefit (Target 1, 2 and 6); Magenta: intermediate information benefit (Target 4, 5 and 7); Green: low information benefit (Target 3 and 8). Optimal path τ^* is obtained given the parameters $w_B = 20, w_J = 1$.

Chapter 8

Conclusion

The objective of the treasure hunt problem is to infer hidden variable(s) or treasure(s) from an available set of measurements that are accessible only through observation cells in a given connectivity graph $\mathcal{G}$. A novel and systematic approach has been proposed to solve this class of problem. Using cell decomposition, the pruning algorithm and the benefit-of-information function presented in this dissertation, a reduced subset of feasible solutions is obtained in the form of a pruned connectivity tree, which can be folded into a decision graph, such as, decision tree or influence diagram. The proposed new approximate cell decomposition approach for robotic sensor path planning accounts for not only the geometry of obstacles but also the geometry of the targets and of the sensor field of view. The keystones of this novel approach is the new concepts of observation cells and C-targets. BN sensor model/formalism is developed for target classification and measurement fusion. The solution of the treasure hunt problem is a nonmyopic global optimal policy that produces the path in $\mathcal{G}$ and the corresponding measurement sequence with the maximum expected observation profit.

The board game of CLUE® is found to be an excellent benchmark for the treasure hunt problem. Therefore, the methodology is implemented to obtain an intelligent computer player that outperforms several human players, as well as a computer player obtained by Bayesian networks only, Q-learning, or constraint satisfaction. The landmine detection and classification application with the crucial characteristics of treasure hunt problem verifies the efficiency of the proposed approach. Information-driven sensor path planning achieves highest path efficiency among shortest path, complete

coverage, random search, and fixed grid strategies.

The methodologies proposed here have many applications in the sensor network community pertaining to target detection, classification and tracking. The recommendations for future work lie in the following directions: consider a moving FOV instead of a fixed FOV with regard to the robot geometry in this dissertation; improve real-time feedback of new sensor measurements; extend the sensor path planning methodology to track and pursue dynamic targets, since the targets considered in this dissertation are static; extend the methodology to multi-sensor multi-platform applications, e.g., multi-platform collaborative demining, while only one platform and one robotic sensor is considered in the demining application in this dissertation; and implement the proposed approach in demining physical experiments.

Appendix A

Theoretic Relationships between Expected Entropy Reduction and Expected discrimination Gain

The proposed EER is defined as:

$$\Delta H(y; M|\mathcal{E}^0) \equiv H(y|\mathcal{E}^0) - \sum_M \left[H(y|\mathcal{E}^0,\ M)P(M|\mathcal{E}^0)\right]. \tag{A.1}$$

It is actually a type of conditional mutual information $I(y; M|\mathcal{E}^0)$.

The expected discrimination gain (EDG) is defined as:

$$\begin{aligned}
&\Delta D(y; M|\mathcal{E}^0) \\
&\equiv \sum_M \left[D(y|\mathcal{E}^0,\ M)P(M|\mathcal{E}^0)\right] - D\left(y|\mathcal{E}^0\right) \\
&= \sum_M \left[P(M|\mathcal{E}^0) \sum_y P(y|\mathcal{E}^0,\ M) \log_2 \frac{P(y|\mathcal{E}^0,\ M)}{P(y)}\right] \\
&\quad - \sum_y P(y|\mathcal{E}^0) \log_2 \frac{P(y|\mathcal{E}^0)}{P(y)} \\
&= \sum_M \{P(M|\mathcal{E}^0)[\sum_y P(y|\mathcal{E}^0,\ M) \log_2 P(y|\mathcal{E}^0,\ M) \\
&\quad - \sum_M P(y|\mathcal{E}^0,\ M) \log_2 P(y)]\} \\
&\quad - [\sum_M P(y|\mathcal{E}^0) \log_2 P(y|\mathcal{E}^0) - \sum_y P(y|\mathcal{E}^0) \log_2 P(y)]
\end{aligned}$$

$$
\begin{aligned}
&= \sum_M \{P(M|\mathcal{E}^0) \sum_y P(y|\mathcal{E}^0,\ M) \log_2 P(y|\mathcal{E}^0,\ M)\} \\
&\quad - \sum_M \{P(M|\mathcal{E}^0) \sum_y P(y|\mathcal{E}^0,\ M) \log_2 P(y)\} \\
&\quad - [\sum_y P(y|\mathcal{E}^0) \log_2 P(y|\mathcal{E}^0) - \sum_y P(y|\mathcal{E}^0) \log_2 P(y)] \\
&= \sum_M \{P(M|\mathcal{E}^0) \sum_y P(y|\mathcal{E}^0,\ M) \log_2 P(y|\mathcal{E}^0,\ M)\} \\
&\quad - \sum_y P(y|\mathcal{E}^0) \log_2 P(y|\mathcal{E}^0) \\
&\quad - \sum_M \{P(M|\mathcal{E}^0) \sum_y P(y|\mathcal{E}^0,\ M) \log_2 P(y)\} \\
&\quad + \sum_y P(y|\mathcal{E}^0) \log_2 P(y) \\
&= \sum_M \{P(M|\mathcal{E}^0) H(y|\mathcal{E}^0,\ M)\} - H(y|\mathcal{E}^0) \\
&\quad - \sum_M \{P(M|\mathcal{E}^0) \sum_y P(y|\mathcal{E}^0,\ M) \log_2 P(y)\} \\
&\quad + \sum_y P(y|\mathcal{E}^0) \log_2 P(y) \\
&= \Delta H(y; M|\mathcal{E}^0) \\
&\quad - \sum_M \{P(M|\mathcal{E}^0) \sum_y P(y|\mathcal{E}^0,\ M) \log_2 P(y)\} \\
&\quad + \sum_y P(y|\mathcal{E}^0) \log_2 P(y) \\
&= \Delta H(y; M|\mathcal{E}^0) - \sum_M \sum_y \{P(M|\mathcal{E}^0) P(y|\mathcal{E}^0,\ M) \log_2 P(y)\} \\
&\quad + \sum_y P(y|\mathcal{E}^0) \log_2 P(y)
\end{aligned}
$$

$$
\begin{aligned}
&= \Delta H(y; M|\mathcal{E}^0) - \sum_M \sum_y \{P(y, M|\mathcal{E}^0) \log_2 P(y)\} \\
&\quad + \sum_y P(y|\mathcal{E}^0) \log_2 P(y) \\
&= \Delta H(y; M|\mathcal{E}^0) - \sum_y P(y|\mathcal{E}^0) \log_2 P(y) + \sum_y P(y|\mathcal{E}^0) \log_2 P(y) \\
&= \Delta H(y; M|\mathcal{E}^0) \qquad \text{(A.2)}
\end{aligned}
$$

It is shown above that theoretically, the proposed EER $\Delta H(y; M|\mathcal{E}^0)$ equals the EDG $\Delta D(y; M|\mathcal{E}^0)$ which was first proposed in [25]. Both of them are actually mutual information.

Appendix B

Properties of Approximate Cell Decomposition in the Presence of Targets

This section analyzes the properties of the cell decomposition presented in Section 4.1.

Property 1 *For $\forall q \in \kappa^u$, if $q \notin \mathcal{RB}_j[\kappa^u]$, then $\mathcal{A}$ avoids collisions with $\mathcal{B}_j$, and if $q \in \mathcal{RT}_i[\kappa^u]$, then $\mathcal{S}$ can make measurements from $\mathcal{T}_i$.*

Proof: It follows from the definition of bounding approximation and bounded approximation that $\mathcal{CB}_j[\kappa^u] \subseteq \mathcal{OCB}_j[\kappa^u] \times \mathcal{I}_u$ and $\mathcal{ICT}_i[\kappa^u] \times \mathcal{I}_u \subseteq \mathcal{CT}_i[\kappa^u]$. For bounding rectangloid approximation, $\mathcal{RB}_j[\kappa^u]$, of $\mathcal{OCB}_j[\kappa^u] \times \mathcal{I}_u$, we have $\mathcal{OCB}_j[\kappa^u] \times \mathcal{I}_u \subseteq \mathcal{RB}_j[\kappa^u]$. For bounded rectangloid approximation, $\mathcal{R'T}_i[\kappa^u]$, of $\mathcal{ICT}_i[\kappa^u] \times \mathcal{I}_u$, we have $\mathcal{R'T}_i[\kappa^u] \subseteq \mathcal{ICT}_i[\kappa^u] \times \mathcal{I}_u$. It follows that $\mathcal{CB}_j[\kappa^u] \subseteq \mathcal{RB}_j[\kappa^u]$ and $\mathcal{R'T}_i[\kappa^u] \subseteq \mathcal{CT}_i[\kappa^u]$, $q \notin \mathcal{RB}_j[\kappa^u]$, then $q \notin \mathcal{CB}_j[\kappa^u]$, since $q \in \kappa^u$ implies $q \notin \mathcal{CB}_j$. Thus, $\mathcal{A}(q) \cap \mathcal{B}_j = \varnothing$, which means that $\mathcal{A}$ avoids collisions with $\mathcal{B}_j$. If $q \in \mathcal{RT}_i[\kappa^u]$, then $q \in \mathcal{CT}_i[\kappa^u]$, and thus $\mathcal{S}(q) \cap \mathcal{T}_i \neq \varnothing$, which means that $\mathcal{S}$ intersects $\mathcal{T}_i$, or in other words $\mathcal{S}$ can take measurements from $\mathcal{T}_i$. $\square$

Property 2 *If $\mathcal{A}$, $\mathcal{S}$, $\mathcal{B}_j$ $(j = 1, \ldots, n)$, and $\mathcal{T}_i$ $(i = 1, \ldots, r)$ are all convex polygons in an Euclidean workspace $\mathcal{W} \subset \mathbb{R}^2$, the time complexity of the approximate cell decomposition in the presence of targets (Section 4.1) is $O((n_\mathcal{B} + n_\mathcal{T})^2)$, where $n_\mathcal{B}$ is the number of edges of all n obstacles, and $n_\mathcal{T}$ is the number of edges of all r targets.*

Proof: If $\mathcal{A}$, $\mathcal{S}$, $\mathcal{B}_j$ $(j = 1, \ldots, n)$, and $\mathcal{T}_i$ $(i = 1, \ldots, r)$ are all convex polygons in $\mathcal{W} \subset \mathbb{R}^2$, then $\mathcal{CB}_j$ $(j = 1, \ldots, n)$ and $\mathcal{CT}_i$ $(i = 1, \ldots, r)$ are also convex [48]. Let

$n_{\mathcal{A}}$ denote the number of edges of $\mathcal{A}$, which can be considered a constant. In Steps (1)-(2) (Section 4.1), $\mathcal{CB}_j[\kappa^u], j = 1, \ldots, n$ and $\mathcal{CT}_i[\kappa^u], i = 1, \ldots, r$ for every every $u = 1, \ldots, \nu$ are computed by first discretizing $\mathcal{I}_u$ into k_u values, with $\gamma_u + l\Delta\theta$ for $0 \leq l \leq k_u$, and $\Delta\theta = (\gamma_{u+1} - \gamma_u)/k_u$. This strategy converts the computation of $\mathcal{CB}_j[\kappa^u]$ and $\mathcal{CT}_i[\kappa^u]$ in a 3D world to the computation of C-obstacles and C-targets in a 2D world at different robot orientations (e.g., imagine $\mathcal{CB}_j[\kappa^u]$ and $\mathcal{CT}_i[\kappa^u]$ as a stack of k_u polygonal regions). The time for computing the 2D C-obstacle $\mathcal{CB}_j[[x_\kappa, x'_\kappa] \times [y_\kappa, y'_\kappa] \times (\gamma_u + l\Delta\theta)]$, for all $j = 1, \ldots, n$, in $\mathcal{W}$ at the robot rotation orientation $(\gamma_u + l\Delta\theta)$ for every l, $0 \leq l \leq k_u$, is $O(n_{\mathcal{B}})$. The time for computing $\mathcal{CT}_i[[x_\kappa, x'_\kappa] \times [y_\kappa, y'_\kappa] \times (\gamma_u + l\Delta\theta)]$, for all $i = 1, \ldots, r$, in $\mathcal{W}$ at the robot rotation orientation $(\gamma_u + l\Delta\theta)$ for every l, $0 \leq l \leq k_u$, is $O(n_{\mathcal{T}})$ [63]. Since k_u is a constant, the times for computing $\mathcal{CB}_j[\kappa^u]$, for all $j = 1, \ldots, n$, and $\mathcal{CT}_i[\kappa^u]$, for all $i = 1, \ldots, r$, are also $O(n_{\mathcal{B}})$ and $O(n_{\mathcal{T}})$, respectively. Based on the numerical approach for computing $\mathcal{CB}_j$ and $\mathcal{CT}_i$,

$$\mathcal{OCB}_j[\kappa^u] = \bigcup_{l=1}^{k_u} \{\mathcal{CB}_j[[x_\kappa, x'_\kappa] \times [y_\kappa, y'_\kappa] \times (\gamma_u + l\Delta\theta)]\} \tag{B.1}$$

$$\mathcal{ICT}_i[\kappa^u] = \bigcap_{l=1}^{k_u} \{\mathcal{CT}_i[[x_\kappa, x'_\kappa] \times [y_\kappa, y'_\kappa] \times (\gamma_u + l\Delta\theta)]\}. \tag{B.2}$$

for every $u = 1, \ldots, \nu$. Thus, the time for computing $\mathcal{OCB}_j[\kappa^u]$ in (B.1) is $O((n_{\mathcal{A}} + c_j)\log(n_{\mathcal{A}} + c_j))$ where c_j is the number of edges of $\mathcal{B}_j$ [64], $\sum_{j=1}^{n} c_j = n_{\mathcal{B}}$. Since this approximate cell decomposition is resolution-complete, it can be assumed that the number of orthogonal rectangloids in $\mathcal{RB}_j[\kappa^u]$ and $\mathcal{R'T}_i[\kappa^u]$ is linearly proportional to the number of edges of B_j and T_i, respectively. Then, the time for generating the bounding rectangloid approximation $\mathcal{RB}_j[\kappa^u]$ of $\mathcal{OCB}_j[\kappa^u] \times \mathcal{I}_u$ (Step (3)) is $O(c_j)$ and for all $j = 1, \ldots, n$, the time complexity is is $\sum_{j=1}^{n} \{O((n_{\mathcal{A}} + c_j)\log(n_{\mathcal{A}} + c_j)) + O(c_j)\} \leq O(n_{\mathcal{B}} \log n_{\mathcal{B}})$. Similarly, the time complexity to obtain $\mathcal{R'T}_i[\kappa^u]$ for all $i = 1, \ldots, r$

(Step (3)) is $O(n_{\mathcal{T}} \log n_{\mathcal{T}})$.

In Step (4) (Section 4.1), the decomposition of $\mathcal{C}^u_{void}$ into non-overlapping rectangloids can be performed as the vertical decomposition in 2D presented in [63], thus it can be carried out in time $O((n_{\mathcal{B}} + n_{\mathcal{T}}) \log(n_{\mathcal{T}} + n_{\mathcal{T}}))$ [63]. Step (5) (Section 4.1) is comprised of two stages. The first stage decomposes $\mathcal{R}'\mathcal{T}_i[\kappa^u] \backslash \{\bigcup_{j=1}^{n} \mathcal{RB}_j[\kappa^u] \cup \bigcup_{l=1,l\neq i}^{r} \mathcal{R}'\mathcal{T}_l[\kappa^u]\}$ into cells from which only one target is observable. And, the second stage decomposes $\mathcal{R}'\mathcal{T}_i[\kappa^u] \cap \bigcup_{l=1,l\neq i}^{r} \mathcal{R}'\mathcal{T}_l[\kappa^u] \setminus \bigcup_{j=1}^{n} \mathcal{RB}_j[\kappa^u]$ into cells from which two or more targets are observable. The time required by the first stage is $O(n_{\mathcal{B}}+n_{\mathcal{T}})$, because convex polygons are characterized by the property that the boundaries of any pair intersect at most two points. Therefore, the complexity of the common exterior is linear in order with respect to the number of polygons' edges, i.e., $n_{\mathcal{B}} + n_{\mathcal{T}}$ [64]. Also, it can be easily to shown that $\mathcal{R}'\mathcal{T}_i[\kappa^u] \backslash \{\bigcup_{j=1}^{n} \mathcal{RB}_j[\kappa^u] \cup \bigcup_{l=1,l\neq i}^{r} \mathcal{R}'\mathcal{T}_l[\kappa^u]\}$ is an orthogonal polygon without holes, since $\mathcal{RB}_j[\kappa^u]$ and $\mathcal{R}'\mathcal{T}_i[\kappa^u]$ (for $\forall i, j$) are connected, and never contain each other by definition. The time for optimally partitioning an orthogonal polygon without holes into the minimum number of rectangloids is linear in the number of edges [65]. It follows that the first stage takes time $O(n_{\mathcal{T}}(n_{\mathcal{B}} + n_{\mathcal{T}}))$, and the second stage also takes time $O(n_{\mathcal{T}}(n_{\mathcal{B}} + n_{\mathcal{T}}))$. It can be concluded that the entire approximate cell decomposition procedure (Section 4.1) takes time $O(n_{\mathcal{B}}) + O(n_{\mathcal{T}}) + O(n_{\mathcal{B}} \log n_{\mathcal{B}}) + O(n_{\mathcal{T}} \log n_{\mathcal{T}}) + O((n_{\mathcal{B}} + n_{\mathcal{T}}) \log(n_{\mathcal{B}} + n_{\mathcal{T}})) + 2 \times O(n_{\mathcal{T}}(n_{\mathcal{B}} + n_{\mathcal{T}})) \leq O((n_{\mathcal{B}} + n_{\mathcal{T}})^2)$, or simply $O((n_{\mathcal{B}} + n_{\mathcal{T}})^2)$. $\square$

Appendix C

Label-Correcting Pruning Algorithm

The connectivity graph with observations $\mathcal{G}$ is provided as a list of integers $\{0, \pm 1, \ldots, N = f\}$ ordered by their absolute value and corresponding to the list of cells $\mathcal{K} = \{\kappa_0, \kappa_1, \ldots, \kappa_{N=f}\}$, such that $x = \pm i$ represents the cell κ_i, and $x \geq 0$ if κ_i is a void cell, or $x < 0$ if κ_i is an observation cell. The adjacency relations in $\mathcal{G}$ are represented by an $N \times N$ symmetric matrix $A = \{a_{ij}\}$, with $a_{ij} = 1$ if κ_i and κ_j are connected in $\mathcal{G}$, and $a_{ij} = 0$ otherwise; $a_{ii} = 0$ to ensure tree growth. Another symmetric matrix of the same dimensions $D = \{d_{ij}\}$ contains the distance metric evaluated for all arcs in $\mathcal{G}$, and the notation $D(k, m)$ is used to refer to the element in its k^{th} row and m^{th} column, or d_{km}.

The output of the pruning algorithm is a tree T_r in the form of two 2-dimensional arrays, TREE and DIST. Every element or node TREE(j, t) consists of a cell index, $x = \pm i$, representing a cell κ_i that is encountered at time index t along the j^{th} tree branch TREE(j, 0) $\rightarrow$ TREE(j, 1) $\cdots \rightarrow$ TREE(j, $t-1$) $\rightarrow$ TREE(j, t) $\rightarrow$ TREE(j, $t+1$) $\rightarrow \cdots$ TREE(j, t_f). The array DIST contains the distance associated with each arc in TREE, namely DIST(j, t) $= d_{ik}$ if TREE(j, t) $= \pm i$ and TREE(j, $t-1$) $= \pm k$. The pruning algorithm compiles these arrays incrementally, beginning with the root κ_0 and growing branches spatially (column-wise) and temporally (row-wise) one node at a time, and pruning branches that are information-equivalent and sub-optimal with respect to distance. Hence, an array DIST$_{tot}$ also is computed such that its $(j, t)^{th}$-element is the total distance between κ_0 and the node in TREE(j, t). VISITED is a list that contains the indices of the nodes that have been considered so far by the algorithm, and DIST$_{short}$ contains the corresponding distance from κ_0

that is the shortest distance for all information-equivalent branches so far.

The above variable structures support the following operations:

- ADJACENT(x, x_{root}, x_s): obtain all of the nodes adjacent to x in $\mathcal{G}$ that are within a distance d_M of x_s, and do not include x_{root},
- CUT(*branch*, t): remove all nodes that follow the last observation cell in *branch* column-wise, and return the time index t of the last observation cell
- GROW(TREE, x, x_g): add the arc $x \rightarrow x_g$ to the tree structure in TREE
- GETOBSERV(*branch*): extracts all observation cells in *branch* and sorts them in ascending ordering number (i.e., absolute value),
- INSERT(LIST, x): adds an item x at the end of the list in LIST,
- PRUNE(TREE, x): for any branch in TREE with x as a leaf cut the branch down to but not including its first joint, which is the last node going forward in time that generates other branches not ending in x (i.e., is repeated along the same column)

Then, the following algorithm produces the connectivity tree T_r associated with $\mathcal{G}$, κ_0, and κ_f, for a parameter d_M that is chosen by the user:

Pruning Algorithm {

procedure $T_r(\mathcal{G}, \kappa_0, \kappa_f, d_M)$

begin

initialize TREE $= \{\kappa_0\}$, and all other variables as empty

while $\neg$ END(TREE) do

$i_x \leftarrow$ index of first, shortest row in TREE that does not end in κ_f

$i_{void} \leftarrow$ index of first observation-free branch of shortest overall distance

x = TREE(i_x,end), x_{root} = TREE(i_x,end-1), x_s = TREE(i_{void},end)

adjacent $\leftarrow$ ADJACENT(x, x_{root}, x_s)

for every node $x_a \in$ adjacent do

x_g = nil

branch$_{new} \leftarrow$ GROW(TREE(i_x, $\cdot$), x, x_a)

distance$_{new}$ = DIST$_{short}(x)$ + D(x, x_a)

if $x_a \notin$ VISITED then

begin

$x_g = x_a$

VISITED $\leftarrow$ INSERT(VISITED, x_a)

DIST$_{short} \leftarrow$ INSERT(DIST$_{short}$, distance$_{new}$)

end;

else if $x_a > 0$ *[*$x_a \in \mathcal{K}_{void}$*]* and distance$_{new}$ < DIST$_{short}(x_a)$ then

begin

TREE $\leftarrow$ PRUNE(TREE, x_a)

$x_g = x_a$

end;

else if $x_a < 0$ *[$x_a \in \mathcal{K}_z$]* then

begin

observ$_{new}$ = GETOBSERV(branch$_{new}$)

$j \leftarrow$ index value that gives OBSERV(j, $\cdot$) = observ$_{new}$

if j = nil then $x_g = x_a$

else *[observ$_{new} \in$ OBSERV]*

begin

(branch$_{old}$, t) $\leftarrow$ CUT(TREE(j, $\cdot$))

if distance$_{new}$ < DIST$_{tot}$(j, t)

begin

TREE $\leftarrow$ PRUNE(TREE, TREE(j, t))

$x_g = x_a$

end;

end;

end;

TREE $\leftarrow$ GROW(TREE, x, x_g)

Update DIST, DIST$_{tot}$, DIST$_{short}$, and OBSERV based on x and x_g

$x_{child} \leftarrow$ INSERT(x_{child}, x_g)

end; *[for loop]*

if x_{child} = nil then TREE $\leftarrow$ PRUNE(TREE, x)

x_{child} = nil

end; *[while loop]*

end; *[procedure]*

}

Appendix D

Properties of Connectivity Tree Obtained by Pruning

The pruning algorithm applies the principle of optimality [47] to paths connecting two nodes κ_0 and κ_f in $\mathcal{G}$. First, we demonstrate that T_r contains the path of shortest overall distance $d_{0f} = d_{f0}$. Since κ_f is fixed, we seek the shortest path from κ_f to κ_0 by means of dynamic programming, working *backwards* from κ_0 to κ_f. At the second-to-last stage, t_1, the admissible nodes $x(t_1)$ are those adjacent to κ_0 and they all are kept by the algorithm, along with their distance, and marked "visited". At t_1, all paths are already optimal because there is only one path to each admissible cell. At step t_2 however the admissible nodes $x(t_2)$ are all cells adjacent to any cell in $x(t_1)$. Suppose a void cell κ_a is revisited, then only the path with the shortest overall distance d^*_{0a} is kept by the pruning algorithm. Thus, at any moment in time $t_0 \leq t_k \leq t_f$ the set of admissible and void nodes $x^*(t_k) > 0$ that are kept in T_r all lie on the path of minimum distance between t_0 and t_k. Suppose $x^*(t_k) = \{\kappa_a, \kappa_b, \kappa_c\}$, then the paths kept are those with the optimal distances d^*_{0a}, d^*_{0b}, and d^*_{0c}, respectively. By the principle of optimality, it follows that, for each of these paths,

$$\begin{aligned} d^*_{fa0} &= d^*_{0a} + d_{af}, \\ d^*_{fb0} &= d^*_{0b} + d_{bf}, \\ d^*_{fc0} &= d^*_{0c} + d_{cf}, \end{aligned}$$

where, d^*_{fa0} corresponds to the path of minimum distance from κ_f to κ_0, through κ_a. Thus, if any of the cells in $x^*(t_k)$, say κ_b, lie on the path of minimum overall distance, that is $d^*_{f0} = d^*_{fb0}$, then the optimal path kept between κ_0 and κ_b also

lies on *the* optimal path between κ_f and κ_0 (which is found when the algorithm terminates). Since the shortest path connecting κ_f to κ_0 is equivalent to the shortest path connecting κ_0 to κ_f, and $d^*_{f0} = d^*_{0f}$, then the pruning algorithm keeps the optimal path, provided all cells are void.

Now suppose some of the cells in $\mathcal{G}$ are observation cells and, thus, are not always eliminated based solely on distance. If an observation cell, $\bar{\kappa}_e$, is visited for the first time, or is revisited through a non-information-equivalent branch ($j = \text{nil}$), then it always is kept regardless of distance. Since this step does not eliminate but only adds paths, it cannot eliminate the overall optimal path, with distance d^*_{0f}. Instead, suppose $\bar{\kappa}_e$ is visited twice by the algorithm: once at time t_i by a path through a node κ_a, and once at time $t_j \geq t_i$ by a path through κ_b that is characterized by a shorter distance, i.e., $d^*_{0e} = d^*_{0be} < d^*_{0ae}$. Then, if the two paths are information equivalent, the observation cell is treated like a void cell and the path $\kappa_0 \cdots \rightarrow \kappa_b \cdots \rightarrow \bar{\kappa}_e$ is kept, such that,

$$d^*_{fe0} = d^*_{0e} + d_{ef}.$$

Thus, if $\bar{\kappa}_e$ lies on shortest path from κ_f to κ_0, then so does the path through κ_b, with distance d^*_{0be}, that is kept by the pruning algorithm. The same argument applies to any subsequent time when $\bar{\kappa}_e$ is revisited through an information-equivalent branch that is shorter than the one stored in TREE. Hence, the branch that remains when the tree is complete is the shortest of all information-equivalent branches connecting κ_0 and κ_f through $\bar{\kappa}_e$.

The final case is that of an observation cell, say $\bar{\kappa}_g$, that is revisited through an information-equivalent branch through a node κ_a. Suppose the existing information-equivalent branch in TREE, through κ_b, contains $\bar{\kappa}_g$ (by definition of information-equivalent branches) but terminates in another cell κ_c. In other words, the two branches are distinct and information equivalent, but the paths connecting κ_0 and

$\bar{\kappa}_g$ along them are not information equivalent. In this case, the shortest branch can still be eliminated. Consider the existing branch up to the last observation cell in it, say $\bar{\kappa}_e$, such that the new branch $\kappa_0 \cdots \rightarrow \kappa_a \cdots \rightarrow \bar{\kappa}_g$ can be compared to the shortest information-equivalent path in the existing branch, namely $\kappa_0 \cdots \rightarrow \kappa_b \cdots \rightarrow \bar{\kappa}_g \cdots \rightarrow \bar{\kappa}_e$ (which has been pruned of the path connecting $\bar{\kappa}_e$ to κ_c, but may still contain κ_b as well as any other nodes anywhere along the path). If $d_{0ge} > d_{0ag}$, then $d_{0ag} < d_{0bg}$ because d_{ge} cannot be negative. By the principle of optimality $d^*_{0g} = d_{0ag}$, and if $\bar{\kappa}_g$ lies on the overall optimal path from κ_f to κ_0, so does the path $\kappa_0 \cdots \rightarrow \kappa_a \cdots \rightarrow \bar{\kappa}_g$. Since the existing path through κ_b to $\bar{\kappa}_g$ is distinct and sub-optimal with respect to the former, then it cannot lie on the shortest path from κ_f to κ_0. Thus, this existing branch can be eliminated because it is information-equivalent and longer in distance than the new branch. Moreover, eliminating this branch does not eliminate another possibly-optimal path, because if such a path went through the existing branch it would go through $\bar{\kappa}_g$, and if $\bar{\kappa}_g$ lies on the optimal path, then by the principle of optimality the optimal path must include $\kappa_0 \cdots \rightarrow \kappa_a \cdots \rightarrow \bar{\kappa}_g$, or $d^*_{f0} = d^*_{0ag} + d_{gf}$.

Finally, through the operation ADJACENT, the pruning algorithm disallows the parent of a node to be its direct child because this branch always leads to a suboptimal path. Consider a branch $\kappa_0 \cdots \rightarrow \kappa_p \rightarrow \kappa_c$ in TREE that can be grown through the nodes adjacent to its last cell κ_c. By definition the parent κ_p always is adjacent to the child node in TREE, in this case κ_c. Also, if κ_p is in TREE it implies that its branch contains the shortest information-equivalent path from κ_0 to κ_p with the shortest distance d^*_{0p}, and $d^*_{f0} = d^*_{0p} + d_{pf}$. Thus, if κ_c is added to the branch along with κ_p as its direct child, producing $\kappa_0 \cdots \rightarrow \kappa_p \rightarrow \kappa_c \rightarrow \kappa_p \cdots \rightarrow \kappa_f$, the resulting total distance is $d^*_{0p} + 2 \cdot d_{pc} + d_{pf} > d^*_{0p} + d_{pc} + d_{pf}$, and thus is sub-optimal even when κ_p lies on the optimal path from κ_f to κ_0. Moreover, since the order of the

measurements is irrelevant (Remark 5.3.2), if κ_p is an observation cell, revisiting it at a later time never adds information value to the path, and the branch remains information-equivalent to itself.

Appendix E

Proof of Theorem 5.3.1

At time t_0, before any measurements are taken, the uncertainty in the hypothesis variable y is the entropy $H(y)$, computed from the prior probability $P(y)$ (eq. 3.2). At time t_1, when the first measurement z_1 is obtained, the uncertainty in y is given by $H(y|z_1)$ and using the result that conditioning reduces entropy [41],

$$H(y|z_1) \leq H(y) \tag{E.1}$$

where, the equality holds if and only if $y \perp z_1$. At time t_2, when a second measurement z_2 is obtained, the chain rule for entropy [41] is applied, such that,

$$H(y, z_1, z_2) = H(y, z_1|z_2) + H(z_2) = H(y|z_2, z_1) + H(z_1|z_2) + H(z_2) \tag{E.2}$$

and

$$H(y, z_1) = H(z_1) + H(y|z_1). \tag{E.3}$$

Then, the conditional entropy at t_2 can be written as,

$$H(y|z_1, z_2) = H(y, z_1|z_2) - H(z_1|z_2) \leq H(y, z_1) - H(z_1|z_2), \tag{E.4}$$

where, the inequality applies because conditioning reduces entropy. Using eq. (E.3), the above inequality is

$$H(y|z_1, z_2) \leq H(y|z_1) + H(z_1) - H(z_1|z_2). \tag{E.5}$$

Furthermore, $H(z_1|z_2) \leq H(z_1)$, thus $[H(z_1) - H(z_1|z_2)] \geq 0$, and it follows from eq. (E.5) that conditioning the probability of y upon both z_1 and z_2 must reduce entropy with respect to conditioning y upon z_1 alone, regardless of the outcome of z_2:

$$H(y|z_1, z_2) \leq H(y|z_1). \tag{E.6}$$

Let z_i and z_j be any two measurements taken at times t_i and t_j, respectively, during the Markov process. Then, by induction,

$$H(y|z_1, \ldots, z_{f-1}) \leq H(y|z_1, \ldots, z_j) \leq H(y|z_1, \ldots, z_i) \leq H(y) \tag{E.7}$$

provided $t_j > t_i$. Thus, the conditional entropy is a decreasing function over T, and when the measurements are not independent (as by our initial assumption) it is a monotonically decreasing function, since the above inequalities all become strict inequality.

Therefore, if we define $H(t_k)$ to be the entropy of the hypothesis variable y conditioned upon all of the measurements obtained up to t_k, the incremental entropy between two subsequent time steps,

$$\Delta H(t_k) \equiv H(t_{k-1}) - H(t_k) = I(y;\ z_k \mid z_{k-1}, z_{k-2}, \ldots, z_1) \geq 0 \tag{E.8}$$

is the reduction in entropy brought about by the latest measurement z_k, and from eq. (E.7) it is always a non-negative quantity. From the definition in eq. (3.4), it can be seen that the incremental entropy between two time steps is the mutual information between y and the latest measurement z_k, given all previous measurements up to z_{k-1}. The incremental entropy is said to be an additive function over time because for any two time instants $t_i, t_j \in T$, with $t_i < t_j$, the total reduction in entropy incurred over the time interval $[t_i, t_j]$ is equal to the sum of the incremental entropies,

$$\begin{aligned}
&\textstyle\sum_{k=i}^{j} \Delta H(t_k) = \Delta H(t_i) + \Delta H(t_{i+1}) + \cdots + \Delta H(t_{j-1}) + \Delta H(t_j) \\
&= H(y|z_1, \ldots, z_{i-1}) - H(y|z_1, \ldots, z_i) + H(y|z_1, \ldots, z_i) - H(y|z_1, \ldots, z_{i+1}) \\
&+ \cdots \\
&+ H(y|z_1, \ldots, z_{j-2}) - H(y|z_1, \ldots, z_{j-1}) + H(y|z_1, \ldots, z_{j-1}) - H(y|z_1, \ldots, z_j) \\
&= H(y|z_1, \ldots, z_{i-1}) - H(y|z_1, \ldots, z_j) = I(y; z_j, \ldots, z_i | z_{i-1}, \ldots, z_1)
\end{aligned} \tag{E.9}$$

that simplifies to the given mutual information because all of the intermediate terms cancel each other out. It also follows that since there are no prior measurements at

time t_0, the total reduction in entropy,

$$\sum_{k=1}^{f-1} \Delta H(t_k) = H(y) - H(y|z_1, \ldots, z_{f-1}) = I(y; z_1, \ldots, z_{f-1}) = B(t_f) \qquad \text{(E.10)}$$

is the reduction in the uncertainty of y due to the knowledge of all of the measurements in the given sequence. $\square$

Appendix F

Proof of Remark 5.3.2

Let $M_k = \{m_\ell, m_j, \ldots, m_l\} \subset M$ denote a subset of i measurements in $M = \{m_1, \ldots, m_r\}$, with $i < r$, that are performed over the time period $(t_0, t_k] \in T$, leading to the sequence $Z_{t_k} = \{z_1 = m_\ell,\ z_2 = m_j,\ \ldots,\ z_i = m_l\}$. Then, the total measurement benefit up to time t_k is,

$$B(t_k) = \sum_{\jmath=1}^{k} \Delta H(t_\jmath) = I(y;\ z_1, z_2, \ldots, z_i) = I(y;\ m_\ell, m_j, \ldots, m_l) \tag{F.1}$$

according to Theorem 5.3.1. Where, M_k and M are unordered sets, while Z_{t_k} is a totally ordered set. Consider a different sequence of the same tests M_k, such that another totally ordered set is defined, for example $Z'_{t_k} = \{z_1 = m_l,\ z_2 = m_\ell,\ \ldots,\ z_i = m_j\}$. Then, the total measurement benefit up to time t_k is:

$$B'(t_k) = \sum_{\jmath=1}^{k} \Delta H(t_\jmath) = I(y;\ z_1, z_2, \ldots, z_i) = I(y;\ m_l, m_\ell, \ldots, m_j) \tag{F.2}$$

From the definition of mutual information (eq. 3.4),

$$\begin{aligned} I(y;\ m_\ell, m_j, \ldots, m_l) &= E_{y,m_\ell,m_j,\ldots,m_l}\left\{\log_2 \frac{P(y \mid m_\ell, m_j, \ldots, m_l)}{P(y)}\right\} \\ &= E_{y,m_l,m_\ell,\ldots,m_j}\left\{\log_2 \frac{P(y \mid m_l, m_\ell, \ldots, m_j)}{P(y)}\right\} \\ &= I(y;\ m_l, m_\ell, \ldots, m_j) \end{aligned} \tag{F.3}$$

where, the expectation $E_{y,m_\ell,m_j,\ldots,m_l}$ is computed by marginalizing its argument multiplied by the joint probability distribution, $P(y, m_\ell, m_j, \ldots, m_l)$. Since both the

marginalization operation and the union of sets, denoted by a comma in the probability distributions, are commutative, the mutual information of y and the sequence Z_{t_k} can be written as in eq. (F.3) above. Thus, $I(y; z_1, z_2, \ldots, z_i) = I(y;\ M_k)$, or $B(t_k) = B'(t_k)$, for any ordered sequence of measurements Z_{t_k} containing the set of measurements M_k. $\square$

Bibliography

[1] C. Hofner and G. Schmidt. Path planning and guidance techniques for an autonomous mobile cleaning robot. *Robotics and Autonomous Systems*, 14:199–212, 1995.

[2] E. U. Acar. Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods. *International Journal of Robotic Research*, 22:7–8, 2003.

[3] C. Kreucher, K. Kastella, and A. Hero. Multi-platform information-based sensor management. In *Proc. of the SPIE Defense Transformation and Network-Centric Systems Symposium*, volume 5820, pages 141–151, Orlando, FL, 2005.

[4] C. Cai and S. Ferrari. Information-driven sensor path planning by approximate cell decomposition. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, in review, 2008.

[5] N. Rao, S. Hareti, W. Shi, and S. Iyengar. Robot navigation in unknown terrains: Introductory survey of non-heuristic algorithms. In *Technical Report ORNL/TM-12410*. Oak Ridge National Laboratory, Oak Ridge, TN, 1993.

[6] N. Rao. Robot navigation in unknown generalized polygonal terrains using vision sensors. *IEEE Transactions on System, Man, and Cybernetics*, 25(6):947–962, 1995.

[7] A. Lazanas and J. C. Latombe. Motion planning with uncertainty - a landmark approach. *Artificial Intelligence*, 76:287–317, 1995.

[8] K. Song and C. C. Chang. Reactive navigation in dynamic environment using a multisensor predictor. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 29(6):870–880, 1999.

[9] Z. Sun and J.H. Reif. On robotic optimal path planning in polygonal regions with pseudo-eucledian metrics. *IEEE Transactions on Systems, Man, and Cybernetics - Part A*, 37(4):925–936, 2007.

[10] X.-C. Lai, S.-S. Ge, and A. Al-Mamun. Hierarchical incremental path planning and situation-dependent optimized dynamic motion planning considering accelerations. *IEEE Transactions on Systems, Man, and Cybernetics- Part A*, 37(6):1541–1554, 2007.

[11] H. Choset. Coverage for robotics: A survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31(1-4):113–126, 2001.

[12] C. Cai and S. Ferrari. A Q-learning approach to developing an automated neural computer player for the board game of CLUE®. In *International Joint Conference on Neural Networks*, Hong Kong, 2008.

[13] R. Siegel. Land mine detection. *IEEE Instrumentation and Measurement Magazine*, 5(4):22–28, 2002.

[14] S. Ferrari, C. Cai, R. Fierro, and B. Perteet. A multi-objective optimization approach to detecting and tracking dynamic targets in pursuit-evasion games. pages 5316–5321, New York, NY, 2007.

[15] D. Culler, D. Estrin, and M. Srivastava. Overview of sensor networks. *Computer*, 37(8):41–49, 2004.

[16] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein. Energy efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *Proc. of the 10^{th} International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-X)*, pages 96–107, San Jose, CA, 2002.

[17] X. Liao and L. Carin. Application of the theory of optimal experiments to adaptive electromagnetic-induction sensing of buried targets. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(8):961–972, 2004.

[18] J. R. Spletzer and C. J. Taylor. Dynamic sensor planning and control for optimally tracking target. *International Journal of Robotics Research*, 22(1):7–20, Jan. 2003.

[19] G. D. Hager and M. Mintz. Computational methods for task-directed sensor data fusion and sensor planning. *International Journal of Robotics Research*, 10:285–313, 1991.

[20] S. Y. Chen and Y. F. Li. Automatic sensor placement for model-based robot vision. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 34(1):393–408, 2004.

[21] S. Y. Chen and Y. F. Li. Vision sensor planning for 3cd model acquisition. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 35(5):894–904, 2005.

[22] E. Gelenbe and Y. Cao. Autonomous search for mines. *European Journal of Operation Research*, 108:319–333, 1998.

[23] M. Qian and S. Ferrari. Probabilistic deployment for multiple sensor systems. In *Proc. of the 12th SPIE Symposium on Smart Structures and Materials: Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems*, volume 5765, pages 85–96, San Diego, 2005.

[24] D.J. Zhu and J.-C. Latombe. New heuristic algorithms for efficient hierarchical path planning. *IEEE Transactions on Robotics and Automation*, 7(1):9–20, 1991.

[25] K. Kastella. Discrimination gain to optimize detection and classification. *IEEE Transactions on Systems, Man, and Cybernetics - Part A*, 27(1):112–116, 1997.

[26] C. Kreucher, K. Kastella, and A. Hero. Sensor management using an active sensing approach. *Signal Processing*, 85:607–624, 2005.

[27] F. Zhao, J. Shin, and J. Reich. Information-driven dynamic sensor collaboration. *IEEE Signal Processing Magazine*, 19:61–72, 2002.

[28] W. Schmaedeke. Information based sensor management. In *Proc. of SPIE Signal Processing, Sensor Fusion, and Target Recognition II*, volume 1955, pages 156–164, Orlando, FL, 1993.

[29] S. Ji, R. Parr, and L. Carin. Nonmyopic multiaspect sensing with partially observable markov decision processes. *IEEE Transactions on Signal Processing*, 55(1):2720–2730, 2007.

[30] C. Cai and S. Ferrari. Comparison of information-theoretic objective functions for decision support in sensor systems. In *Proc. of American Control Conference*, pages 63–133, New York, NY, 2007.

[31] C. Cai and S. Ferrari. On the development of an intelligent computer player for CLUE®: a case study on preposterior decision analysis. In *Proc. of American Control Conference*, pages 4350– 4355, Minneapolis, MN, 2006.

[32] S. Russell and P. Norvig. *Artificial Intelligence A Modern Approach*. Prentice Hall, Upper Saddle River, NJ, 2003.

[33] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer-Verlag, 2001.

[34] R. L. Winkler. *Bayesian Inference and Introduction*. Probabilistic Publishing, 2003.

[35] S. Ferrari and A. Vaghi. Demining sensor modeling and feature-level fusion by bayesian networks. *IEEE Sensors*, 6:471–483, 2006.

[36] *Explosive and Ordnance and Disposal and (EOD) and Technicians*. [ORDATA Online]. Avaliable: http://maic.jmu.edu/ordata/mission.asp, 2006.

[37] A. P. Dempster, N. M. Laird, and D. B. Rubin. A generalization of bayesian inference. *Journal of the Royal Statistical Society: Series B*, 39:1–39, 1968.

[38] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, NJ, 1976.

[39] M.L. Hernandez, T. Kirubarajan, and Y. Bar-Shalom. Multisensor resource deployment using posterior cramer-rao bounds. *IEEE Trans. Aerospace Electron. Engng.*, 40(2):399C416, 2004.

[40] B. Ristic, A. Farina, and M. Hernandez. Cramer-rao lower bound for tracking multiple targets. In *Proc. of the IEE, Radar, Sonar and Navigation*, volume 151, page 129C134, 2004.

[41] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley and Sons, Inc., 1991.

[42] K. Košmelj, P. Kalan, and A. Cedilnik. Modification of the fisher's information measure to optimize a sampling design. In *Proc. of the 24th Int. Conf. Information Technology Interfaces (ITI)*, pages 97–101, Cavtat, Croatia, 2002.

[43] A. Rényi. On measures of entropy and information. In *the Fourth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, page 547C561, 1961.

[44] J. Principe, D. Xu, and J. Fisher. Information theoretic learning. In S. Haykin, editor, *Unsupervised Adaptive Filtering*, pages 265–319. Wiley, 1999.

[45] M. D. Plumbley. On information theory and unsupervised neural networks. In *Technical Report CUED/F-INFENG/TR. 78*. Cambridge Unversity Engineering Department, UK, 1991.

[46] S. Ferrari and C. Cai. Information-driven search strategies in the board game of CLUE®. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, in review, 2008.

[47] R. E. Bellman and S. E. Dreyfus. *Applied Dynamic Programming*. Princeton University Press, Princeton, NJ, 1962.

[48] J. C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[49] B. Grünbaum. *Convex Polytopes*. Wiley-Interscience, New York, 1967.

[50] L. C. Polymenakos, D. P. Bertsekas, and J. N. Tsitsiklis. Implementation of efficient algorithms for globally optimal trajectories. *IEEE Transactions on Automatic Control*, 43(2):278–283, 1998.

[51] D. P. Bertsekas. A simple and fast label correcting algorithm for shortest paths. *Networks*, 23(8):703–709, 1993.

[52] E. B. Baum and W. D. Smith. A bayesian approach to relevance in game playing. *Artificial Intelligence*, 97:195–242, 2003.

[53] M. I. Jordan. *Learning in Graphical Models*. MIT Press, 1998.

[54] S. L. Lauritzen and D. Nilsson. Representing and solving decision problems with limited information. *Management Science*, 47(9), 2001.

[55] E. F. Krause. *Taxicab geometry: an adventure in non-Euclidean geometry*. Dover, New York, 1986.

[56] K. Murphy. *How To Use Bayes Net Toolbox*. [Online]. Avaliable: http://www.ai.mit.edu/ murphyk/Software/BNT/bnt.html, 2004.

[57] J. MacDonald. *Alternatives for Landmine Detection*. Rand Publications, 2003.

[58] L. R. Pasion, Billings S. D., and D. W. Oldenburg. Evaluating the effects of magnetic soils on tem measurements for uxo detection. In *Proc. of the 72nd Annual Meeting of the Society of Exploration Geophysicists*, pages 1428–1431, Salt Lake City, UT, 2002.

[59] R. Van Dam, B. Borchers, J. Hendrickx, and S. Hong. Soil effects on thermal signatures of buried nonmetallic landmines. In *Detection and remediation tech-*

nologies for mines and minelike targets VIII, Proc. of the SPIE, volume 5089, pages 1210–1218.

[60] A. Vaghi. *Sensor Management by a Graphical Model Approach*. Laurea Thesis, Politecnico di Milano, 2004.

[61] J.K. Paik. Image processing-based mine detection techniques using multiple sensors: A review. *Subsurface Sensing Technologies and Applications, An International Journal*, 3:203–252, 2002.

[62] M. Qian and S. Ferrari. Probabilistic deployment for multiple sensor systems. In *Proc. of the 12th SPIE Symposium on Smart Structures and Materials: Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems*, volume 5765, pages 85–96, San Diego, 2005.

[63] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[64] K. Kedem, R. Livne, J. Pach, and M. Sharir. On the union of jordan regions and collision-free translational motion amidst polygonal obstacles. *Discrete Computational Geometry*, 1(1):59–71, 1986.

[65] W.-T. Liou, J.-J.-M. Tan, and R.C.T. Lee. Minimum partitioning simple rectilinear polygons in O(nloglogn)-time. In *Proc. of 5th Annual ACM Symposium on Computational Geometry*, pages 344–353, Saarbrüchen, Germany, 1989.

Biography

Chenghui Cai was born in August 1978 in a small village of Hunan Province located in the south of China. Although the life was hardscrabble, he grew up very happily with great care from the whole family, and the childhood gave him not only an intimate acquaintance with the ordinary people but also a keen sense of obligation. He was a naughty student with just OK grade until his eleventh year. In this year, the beginning of his fifth grade in the element school was the turning point of his life, because he met one of the best teachers. His love of study and curiosity of knowledge was strongly inspired. Thereafter, there was no holding him, and he entered Tsinghua University, Beijing, China. In the first two years of his college, he cared a lot about his class grades, and he finally succeeded to realize that the scores are almost useless, at least to him. He began to be absent often in classes and learn a lot outside the classes. He started to have, and still has, strong interests in many subjects, such as philosophy, history, politics, religions, psychology, literature, and biographies of some great scientists. Fortunately, he kept his interests in mathematics, physics and all kinds of mechanics; otherwise, he might not be able to finish a dissertation in engineering today. He also participated in many competitions and got some awards.

His research experience began during his undergraduate program at Tsinghua University, Beijing, China when he worked on a team project of obstacle-avoiding autonomous wheeled mobile robot. He continued his research as a master's student at Tsinghua University working on the power management of a humanoid robot. In his Ph.D. study, he has covered a wide variety of research subjects ranging from robotics, information-driven sensor planning, adaptive dynamic programming, information fusion, computational intelligence to Bayesian learning, inference and decision. His current research interests lie in: design and analysis of algorithms for active

sensing, learning, decision-making under uncertainty, and computational intelligence; investigation of theory and properties of neural and probabilistic networks; development of optimal planning and control techniques with applications to robotic sensors, intelligent surveillance, etc.; and engineering applications of Bayesian inference and decision to statistical modeling from data. So far, he has submitted four papers and published fifteen.