ESCOLA POLITÉCNICA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

DOUTORADO EM CIÊNCIA DA COMPUTAÇÃO

RAMON FRAGA PEREIRA

GOAL RECOGNITION OVER IMPERFECT DOMAIN MODELS

Porto Alegre

2020

PÓS-GRADUAÇÃO - *STRICTO SENSU*

Pontifícia Universidade Católica
do Rio Grande do Sul

PONTIFICAL CATHOLIC UNIVERSITY OF RIO GRANDE DO SUL
SCHOOL OF TECHNOLOGY
COMPUTER SCIENCE GRADUATE PROGRAM

GOAL RECOGNITION OVER
IMPERFECT DOMAIN MODELS

RAMON FRAGA PEREIRA

Thesis presented as partial requirement for obtaining the degree of Ph. D. in Computer Science at Pontifical Catholic University of Rio Grande do Sul.

Advisor: Prof. Prof. Dr. Felipe Meneguzzi (PUCRS)
Co-Advisor: Prof. Dr. Miquel Ramírez (University of Melbourne)

Porto Alegre
2020

Ficha Catalográfica

Ramon Fraga Pereira

GOAL RECOGNITION OVER
IMPERFECT DOMAIN MODELS

This Thesis has been submitted in partial fulfillment of the requirements for the degree of Doctor of Computer Science, of the Graduate Program in Computer Science, School of Technology of the Pontifical Catholic University of Rio Grande do Sul.

Sanctioned on February 20th, 2020.

COMMITTEE MEMBERS:

Prof. Dra. Eva Onaindia (Universitat Politècnica de València)

Prof. Dr. Sebastian Sardina (Royal Melbourne Institute of Technology)

Prof. Dra. Renata Vieira (PPGCC/PUCRS)

Prof. Dr. Miquel Ramírez (University of Melbourne - Co-advisor)

Prof. Dr. Felipe Meneguzzi (PPGCC/PUCRS - Advisor)

GOAL RECOGNITION OVER IMPERFECT DOMAIN MODELS

ABSTRACT

Goal recognition is the problem of recognizing the intended goal of autonomous agents or humans by observing their behavior in an environment. Over the past years, most existing approaches to goal and plan recognition have been ignoring the need to deal with imperfections regarding the domain model that formalizes the environment where autonomous agents behave. In this thesis, we introduce the problem of goal recognition over *imperfect domain models*, and develop solution approaches that explicitly deal with two distinct types of *imperfect domains models*: (1) *incomplete discrete domain models* that have *possible*, rather than *known*, preconditions and effects in action descriptions; and (2) *approximate continuous domain models*, where the transition function is *approximated* from past observations and not well-defined. We develop novel goal recognition approaches over *imperfect domains models* by leveraging and adapting existing recognition approaches from the literature. Experiments and evaluation over these two types of *imperfect domains models* show that our novel goal recognition approaches are accurate in comparison to baseline approaches from the literature, at several levels of observability and imperfections.

RECONHECIMENTO DE OBJETIVOS EM DOMÍNIOS IMPERFEITOS

RESUMO

Reconhecimento de objetivos é definido como a tarefa de reconhecer o objetivo o qual agentes autônomos ou humanos visam alcançar a partir de observações de seu comportamento em um ambiente. Nos últimos anos, a grande maioria das abordagens para reconhecimento de objetivos e planos tem ignorado a necessidade de lidar com imperfeições em modelos de domínios que formalizam o ambiente onde agentes autônomos planejam e agem para alcançar seus objetivos. Nesta tese, introduzimos o problema de reconhecimento em *modelos de domínio com imperfeições*, e desenvolvemos abordagens que explicitamente lidam com dois tipos distintos de *modelos de domínio com imperfeições*: (1) *modelos de domínio discretos e incompletos*, onde as partes incompletas do domínio são anotadas na descrição das ações, usando possíveis precondições e efeitos na descrição das ações; e (2) *modelos de domínio contínuos aproximados*, onde a função de transição não é bem definida e é *aproximada* a partir de observações passadas. As abordagens que desenvolvemos são inovadoras e exploram técnicas conhecidas da literatura de . Experimentos e avaliações mostram que as abordagens que desenvolvemos nesta tese são acuradas quando comparadas a outras abordagens base da literatura, sob diversos níveis de observabilidade e imperfeições.

Palavras Chave: Reconhecimento de Objetivos, Reconhecimento de Planos, Planejamento Automatizado, Dominíos Incompletos, Dominíos Aproximados, Pontos de Referência.

LIST OF ACRONYMS

AUV – Autonomous Underwater Vehicle

AAAI – Association for the Advancement of Artificial Intelligence

IJCAI – International Joint Conference on Artificial Intelligence

ECAI – European Conference on Artificial Intelligence

ICAPS – International Conference on Automated Planning and Scheduling

AAMAS – International Conference on Autonomous Agents and Multi-Agent Systems

IJCNN – International Joint Conference on Neural Networks

PAIR – Workshop on Plan, Activity, and Intent Recognition

AI – Artificial Intelligence

PRAP – Plan Recognition as Planning

STRIPS – Stanford Research Institute Problem Solver

PDDL – Planning Domain Definition Language

FHOC – Finite-Horizon Optimal Control

LQR – Linear–Quadratic Regulator

DNN – Deep Neural Network

RELU – Rectified Linear Unit

RDDL – Relational Dynamic Influence Diagram Language

MILP – Mixed-Integer Linear Program

BLP – Binary Linear Programming

LIST OF FIGURES

LIST OF SYMBOLS

LIST OF TABLES

CONTENTS

1.   INTRODUCTION

Goal Recognition is the problem of discerning the intentions of autonomous agents or humans, given a sequence of observations as evidence of their behavior in an environment, and a domain model describing how the observed agents generate such behavior to achieve their goals [86]. Recognizing goals is important in several applications, especially for monitoring and anticipating agent behavior in an environment, including crime detection and prevention [26], monitoring activities in elder-care [25], recognizing plans in educational environments [91] and exploratory domains [52], and traffic monitoring [76], among others [26, 31, 53, 3].

Existing approaches to solving goal and plan recognition problems vary on the type of domain model used to describe the behavior of the observed agents, or plan generation, as well as the level of observability and noise in the observations used as evidence for recognizing goals and plans [88, Chapter 1]. According to the literature of goal and plan recognition [88, Chapter 1], the most used types of domain models for recognizing goals and plans are *plan–libraries* and *planning domain theories*. Plan–library based approaches to goal and plan recognition have shown to be very fast and accurate in several levels of observability [6, 7, 55, 54]. However, formalizing plan–libraries is usually laborious and time-consuming, and requires a substantial amount of domain knowledge to represent the set of possible plans for achieving the goals. In contrast, recognition approaches that employ the use of planning domain theory and planning techniques have gradually relaxed such requirements [77, 78], resulting in approaches that are very efficient and require much less domain knowledge [40, 19, 87, 70].

A key limitation of most existing approaches to goal and plan recognition is that they fail to deal with incomplete and/or inaccurate domain information available in the domain model. Regardless of the type of domain model formalism used to describe the observed agent's behavior, most recognition approaches assume that the domain model is complete and correct, restricting their direct application to more realistic scenarios in which both the domain model and the observations have imperfect information. Specifically, realistic scenarios have two potential sources of imperfect information. The first stems from imperfections in domain models, especially when such models come from learning processes, approximated from data, or have unknown properties in the action descriptions, resulting in domain models that are not fully accurate due to such imperfections in their description, i.e., *imperfect domain models*. The second stems from ambiguity with respect to the observations, namely, on how imperfect sensor

data report features and properties of the world (e.g., state properties), and on how actions performed by observed agents are realized in the environment.

While there are few research efforts on goal and plan recognition for dealing with *imperfect domain models*, over the past few years the planning community has been addressing this problem in at least two fronts. First, with respect to planning over *incomplete discrete domain models*, in which the planning domain model has annotations to specify what is *unknown* in the model. Weber and Bryce [96], and Nguyen et al. [61, 60] have addressed this problem by developing efficient heuristic approaches for use with well-known and new automated planners. As for the second, we single out the task of planning over *approximate continuous domain models*. Most recently, Say et al. [85] developed an automated planner that can cope with *approximate hybrid mixed discrete-continuous domain models* that are learned from data. Subsequently, Wu, Say, and Sanner [97] developed a planner that relies on modern learning techniques over the same settings proposed by Say et al. [85].

Notwithstanding these developments in *Automated Planning* algorithms, comparatively little effort has been made in the goal and plan recognition community to address this particular problem. However, these developments motivate the key research questions of this thesis, specifically: *Is it possible to recognize goals both quickly and accurately over imperfect domain models? Is it possible to recognize goals over domain models that are approximated from data? Are the current recognition approaches able to remain accurate without any modification/adaption for recognizing goals over imperfect domains?* In this thesis, we aim to address these questions by bringing the problem of goal recognition closer to more realistic scenarios. To do so, we introduce the problem of goal recognition over *imperfect domain models*. To solve this problem, we develop novel goal recognition approaches that can cope with *imperfect domain models*. More specifically, the approaches we develop in this thesis deal explicitly with two types of *imperfect domain models*: *incomplete discrete domain models* and *approximate continuous domain models*. Thus, the main contributions of this thesis are twofold.

Recognizing goals over incomplete discrete domain models: The first contribution of this thesis is regarding the task of goal recognition over *incomplete discrete domain models*. We formalize the problem of goal recognition in *incomplete discrete domain models* by combining the standard formalization of Ramírez and Geffner [77, 78] for goal recognition as planning, and that of Nguyen et al. [61, 60] for planning in incomplete domain models. The formalization of incomplete domains introduced in [61, 60] allows the use of annotations in the domain model to specify what is *incomplete* and *unknown* in the model. Such formalization specifies

the incomplete part of the domain model by using *possible* preconditions and effects in the description actions to specify what is unknown in the domain model. For recognizing goals over incomplete domain models, we develop recognition heuristics that refrain from the use of automated planners during the recognition process. Specifically, we develop novel approaches by enhancing well-known recognition heuristics from the literature [70] that rely on the concept of *landmarks*, and deal explicitly with incomplete domain models. In *Automated Planning*, landmarks are states (or actions) that must be achieved (or executed) to achieve a goal from an initial state [35]. To extract landmarks in incomplete domains and enhance such heuristics, we introduce new notions of landmarks, and we develop a landmark extraction algorithm, adapted from [35]. We evaluate our enhanced recognition heuristics using new datasets constructed by modifying an existing dataset [66] of planning–based goal recognition problems. We have built these new datasets by removing information from the complete domain model and annotating them with *possible* preconditions and effects, which comprise the incomplete part of the domain model. We perform an ablation study to show and understand the effect of the new notions of landmarks on recognition performance. Experiments and evaluation show that our enhanced recognition approaches are fast and accurate for recognizing goals in large and non-trivial incomplete domain models at most levels of domain incompleteness when compared to the non-enhanced (original) recognition approaches in [70].

Recognizing goals over approximate continuous domain models: As for the second contribution of this thesis, we introduce the problem of goal recognition over *approximate continuous domain models*. We develop novel recognition approaches over this setting by leveraging existing work on domain model acquisition via learning techniques for *Hybrid Planning* [85], and adapt well-known probabilistic approaches to goal recognition [78, 39] in order to analyze how prediction errors from the acquired model impact on recognition. Specifically, we use the learning technique proposed by Say et al. [85] to approximate the transition function of continuous domain models, and obtain *approximate continuous domain models*, also named as *nominal models* [47] by the literature of *Control* [13]. We evaluate the recognition approaches over *nominal models* empirically in complex recognition datasets that we built by using three benchmark domains based on the *constrained* Linear–Quadratic Regulator (LQR) problem [9], and two non-linear navigation domains proposed by Say et al. in [85], with increasing dimensions of state and action spaces.

The contributions above can be used to build different types applications in realistic scenarios, such as: online and offline goal recognition applied video streams [31], in which the

domain model can be generated automatically from the video frames [3]; learn the behavior of Autonomous Underwater Vehicles (AUVs) from data [20], and recognize their intended scientific missions based on their interactions in the environment; among others. Thus, the work developed in this thesis brings the task of goal recognition closer to more realistic scenarios, not only by recognizing goals even when the available discrete models are imperfect, but also by performing the recognition task over continuous domains with approximate transition functions.

1.1 Overview of Research Contribution

Throughout the course of our Ph.D. research over the last four years, we have published our contributions as we developed and evaluated them. Particularly, the main contributions that underpin this thesis have been published in five conferences and one journal, as follows.

- Ramon Fraga Pereira, Nir Oren, and Felipe Meneguzzi. *Landmark-Based Approaches for Goal Recognition as Planning* [72]. In Artificial Intelligence, Volume 279, 2020;

- Ramon Fraga Pereira, Mor Vered, Felipe Meneguzzi, and Miquel Ramírez. *Online Probabilistic Goal Recognition over Nominal Models* [75]. In Proceedings of the 28th International Joint Conferences on Artificial Intelligence (IJCAI), 2019;

- Ramon Fraga Pereira, André Grahl Pereira, and Felipe Meneguzzi. *Landmark-Enhanced Heuristics for Goal Recognition in Incomplete Domain Models* [74]. In Proceedings of the 29th International Conference on Automated Planning and Scheduling (ICAPS), 2019;

- Ramon Fraga Pereira and Felipe Meneguzzi. *Goal Recognition in Incomplete Domain Models* [67]. In Proceedings of the 32nd Association for the Advancement of Artificial Intelligence (AAAI)[1], 2018;

- Ramon Fraga Pereira, Nir Oren, and Felipe Meneguzzi. *Landmark-Based Heuristics for Goal Recognition* [70]. In Proceedings of the 31st Association for the Advancement of Artificial Intelligence (AAAI), 2017; and

- Ramon Fraga Pereira and Felipe Meneguzzi. *Landmark-Based Plan Recognition* [65]. In Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI), 2016.

---

[1]This paper has been published as a student abstract at AAAI in 2018, and was among the top-ten best student papers and selected as finalist for the 3-minute presentation contest.

We have also published contributions which, while not directly claimed as part of this thesis, are nevertheless closely related to our contributions.

- Ramon Fraga Pereira, Nir Oren, and Felipe Meneguzzi. *Using Sub-Optimal Plan Detection to Identify Commitment Abandonment in Discrete Environments* [73]. In ACM Transactions on Intelligent Systems and Technology, Volume 11, 2020;

- Leonardo Amado, Ramon Fraga Pereira, Joao Paulo Aires, Mauricio Cecílio Magnaguagno, Roger Granada, Gabriel Paludo Licks, and Felipe Meneguzzi. *LatRec: Recognizing Goals in Latent Space* [2]. Demonstration at the 29th International Conference on Automated Planning and Scheduling (ICAPS), 2019;

- Leonardo Amado, Ramon Fraga Pereira, Joao Paulo Aires, Mauricio Cecílio Magnaguagno, Roger Granada, and Felipe Meneguzzi. *Goal Recognition in Latent Space* [3]. In Proceedings of the International Joint Conference on Neural Networks (IJCNN), 2018;

- Mor Vered, Ramon Fraga Pereira, Mauricio Cecílio Magnaguagno, Gal A. Kaminka, and Felipe Meneguzzi. *Towards Online Goal Recognition Combining Goal Mirroring and Landmarks* [93]. In Proceedings of the 17th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), 2018;

- Ramon Fraga Pereira and Felipe Meneguzzi. *Goal Recognition in Incomplete STRIPS Domain Models* [68]. In the AAAI 2018 workshop on Plan, Activity, and Intent Recognition (PAIR), 2017;

- Roger Granada, Ramon Fraga Pereira, Juarez Monteiro, Rodrigo Barros, Duncan Ruiz, and Felipe Meneguzzi. Hybrid Activity and Plan Recognition for Video Streams [31]. In the AAAI 2017 workshop on Plan, Activity, and Intent Recognition (PAIR), 2017;

- Ramon Fraga Pereira, Nir Oren, and Felipe Meneguzzi. *Monitoring Plan Optimality using Landmarks and Domain-Independent Heuristics* [71]. In the AAAI 2017 workshop on Plan, Activity, and Intent Recognition (PAIR), 2017; and

- Ramon Fraga Pereira, Nir Oren, and Felipe Meneguzzi. *Detecting Commitment Abandonment by Monitoring Sub-Optimal Steps during Plan Execution* [69]. In Proceedings of the 16th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), 2017.

1.2   Thesis Outline

We organized this thesis as follows. In Chapter 2, we provide the relevant background to this thesis[2], revisiting key concepts and terminologies that are essential to understand our contributions. After that, in Chapter 3, we describe a new problem formulation for recognizing goals over incomplete discrete domain models, a landmark extraction algorithm over incomplete domains, along with a set of new notions of landmarks, and develop two landmark–enhanced heuristics that can cope with incomplete domain models. In Chapter 4, we introduce a novel problem formulation for goal recognition over *nominal models*, and describe our solution approaches over this recognition setting. In Chapter 5, we survey the literature and present the related work on recognition under incomplete information, goal and plan recognition as planning, and recent work on planning over *imperfect domain models*. Finally, in Chapter 6, we conclude this thesis by addressing our main contributions, open issues and limitations of our proposed recognition approaches, as well as future avenues regarding the proposed approaches in this thesis.

---

[2]Thus, the reader familiar with *Planning* and *Goal Recognition* may safely skip Chapter 2.

2.   BACKGROUND AND REPRESENTATION

This thesis stands at the intersection of two fields of *Artificial Intelligence* (AI), specifically, *Automated Planning* and *Goal Recognition*, and addresses some of the basic notations and concepts of *Control Theory*. Thus, in this chapter, we present the essential background for understanding the contributions of this thesis. In Section 2.1, we review the background on *Classical Planning* terminology. In Section 2.2, we present the terminology of planning over incomplete domain models. After, in Section 2.3, we describe the concept of *landmarks* in *Automated Planning*, and how we exploit and build some of our recognition approaches using landmarks. Then, in Section 2.4, we describe the task of *Goal and Plan Recognition as Planning* (PRAP). Finally, we conclude this chapter, in Section 2.5, by presenting the terminology of *Control Theory* we use to formalize *Optimal Control* problems.

2.1   Classical Planning

Planning is the problem of finding a sequence of actions (i.e., plan) that achieves a particular goal state from an initial state [82]. Such problem can be seen as a directed graph, whose nodes represent states, edges represent the transition between states (caused by applying actions), and the solution is a path between two particular nodes (i.e., initial state and goal state) in this directed graph. In this thesis, we adopt the terminology from Ghallab et al. [29] to represent states and actions in planning domains and problems. First, we define a *state and its predicates* in the environment as Definition 1.

Definition 1 (**Predicates and State**). *A predicate is denoted by an n-ary predicate symbol p applied to a sequence of zero or more terms (τ_1, τ_2, ..., τ_n) – terms are either constants or variables. We refer to grounded predicates that represent logical values according to some interpretation as facts, which are divided into two types: positive and negated facts, as well as constants for truth ($\top$) and falsehood ($\bot$). A state S is a finite set of positive facts f that follows the closed world assumption so that if $f \in S$, then f is true in S. We assume a simple inference relation $\models$ such that $S \models f$ iff $f \in S$, $S \not\models f$ iff $f \notin S$, and $S \models f_1 \wedge ... \wedge f_n$ iff $\{f_1, ..., f_n\} \subseteq S$.*

A *planning domain model* aims to describe the environment dynamics through the specification of operators, using a limited first-order logic representation (as we defined above in Definition 1) to define schemata for state-modification actions, as follows in Definition 2.

Definition 2 (**Operator and Action**). *An operator op is represented by a triple $\langle name(op), pre(op), eff(op) \rangle$: $name(op)$ represents the description or signature of op; $pre(op)$ describes the preconditions of op, a set of predicates that must exist in the current state for op to be executed; $eff(op)$ represents the effects of op. These effects are divided into $eff^+(op)$ (i.e., an add-list of positive predicates) and $eff^-(op)$ (i.e., a delete-list of negated predicates). An action a is a ground operator instantiated over its free variables.*

We say that an action a is applicable to a state S if and only if $S \models pre(a)$, and generates a new state S' such that:

$$S' := (S \cup eff^+(a))/eff^-(a) \tag{2.1}$$

Thus, by following the notation of *predicates*, *states*, *operator*, and *actions* in Definitions 1 and 2, we formally define a *planning domain model* in Definition 3.

Definition 3 (**Planning Domain**). *A planning domain definition Ξ is represented by a pair $\langle \mathcal{F}, \mathcal{A} \rangle$, which specifies the knowledge of the domain, and consists of:*

- *A finite set of facts $\mathcal{F}$, i.e., a set of ground instantiated predicates, defining the environment state properties; and*

- *A finite set of actions $\mathcal{A}$, which is technically a set of ground instantiated operators, representing the actions that can be performed in the environment.*

A *planning instance*, comprises both a *planning domain* and the elements of a *planning problem*, describing a finite set of *objects* of the environment, the *initial state*, and the *goal state* which an agent wishes to achieve, as formalized in Definition 4.

Definition 4 (**Planning Instance**). *A planning instance Π is represented by a triple $\langle \Xi, \mathcal{I}, G \rangle$, and consists of:*

- *$\Xi = \langle \mathcal{F}, \mathcal{A} \rangle$ is the domain definition, where $\mathcal{F}$ is the set of facts, and $\mathcal{A}$ is the set of actions;*

- $\mathcal{I} \subseteq \mathcal{F}$ *is the initial state specification, which is defined by specifying the value for all facts in the initial state; and*

- $G \subseteq \mathcal{F}$ *is the goal state specification, which represents a desired state to be achieved.*

A *plan* is the solution of a *planning instance*, as formalized in Definition 5.

Definition 5 (Plan). *A plan π for a planning instance $\Pi = \langle \Xi, \mathcal{I}, G \rangle$ is a sequence of actions $\langle a_1,\ a_2,\ ...,\ a_n \rangle$ that modifies the initial state $\mathcal{I}$ into a state $S \models G$ in which the goal state G holds by the successive execution of actions in a plan π. A plan π^* with length $|\pi^*|$ is optimal if there exists no other plan π' for Π such that $\pi' < \pi^*$.*

While instantiated actions have an associated cost, we take the assumption from *Classical Planning* that this cost is 1 for all instantiated actions. Therefore, a plan π is considered *optimal* if its cost, and thus length, is *minimal*.

Modern classical planners use a variety of heuristics to efficiently explore the search space of planning domains by estimating the cost to achieve a specific goal [29]. In *Classical Planning*, this estimate is often the number of actions to achieve the goal state from a particular state. For ease of explanation, we describe our planning–based techniques assuming a uniform action cost $c(a) = 1$ for all $a \in A$, but this is easily generalizable. Thus, the cost for a plan $\pi = [a_1, a_2, ..., a_n]$ is $c(\pi) = \Sigma c(a_i)$.

Heuristics provide no guarantees about the accuracy of their estimations, however, when a heuristic never overestimates the cost to achieve a goal, it is called *admissible* and guarantees optimal plans for certain search algorithms. A heuristic $h(s)$ is admissible if $h(s) \leq h^*(s)$ for all states, where $h^*(s)$ is the optimal cost to the goal state from state s. Heuristics that overestimate the cost to achieve a goal are called *inadmissible*.

2.1.1   STRIPS Domain Models

Classical planning representations often separate the definition of the initial state ($\mathcal{I}$) and goals state (G) as part of a planning problem to be used together with a planning domain model Ξ (Definition 3), such as STRIPS [21] and PDDL [51]. We define a STRIPS domain model over typed variables as $\mathcal{D} = \langle \mathcal{R}, \mathcal{O} \rangle$, where: $\mathcal{R}$ is a set of predicates with typed variables. Grounded predicates represent logical values according to some interpretation as facts, which are divided into two types: positive and negated facts, as well as constants for truth ($\top$) and

falsehood ($\perp$); $\mathcal{O}$ is a set of operators $op = \langle pre(op), eff(op) \rangle$, where $eff(op)$ can be divided into positive effects $eff^+(op)$ (the add list) and negative effects $eff^-(op)$ (the delete list). An operator op with all variables bound is called an action and allows state change. An action a instantiated from an operator op is applicable to a state S iff $S \models pre(a)$ and results in a new state S' such that $S' := (S/eff^-(a)) \cup eff^+(a)$.

A STRIPS planning problem within $\mathcal{D}$ over a set of typed objects Z is defined as $\mathcal{P} = \langle \mathcal{F}, \mathcal{A}, \mathcal{I}, G \rangle$, where: $\mathcal{F}$ is a set of facts (instantiated predicates from $\mathcal{R}$ and Z); $\mathcal{A}$ is a set of instantiated actions from $\mathcal{O}$ and Z; $\mathcal{I}$ is the initial state ($\mathcal{I} \subseteq \mathcal{F}$); and G is the goal state, which represents a desired state to be achieved. A plan π for a planning problem $\mathcal{P}$ is a sequence of actions $\langle a_1, a_2, ..., a_n \rangle$ that modifies the initial state $\mathcal{I}$ into a state $S \models G$ in which the goal state G holds by the successive execution of actions in a plan π.

2.2   Planning in Incomplete Domain Models

Most planning algorithms and heuristics assume that the domain model is complete and correct, relaxing the need to deal with incomplete domain information. We argue that this assumption may be too strong for dealing with more realistic domains. Moreover, the effort of domain knowledge engineering that is required to model a complete and correct planning domain model can be laborious and substantial because of human error and/or lack of domain knowledge from the modeler [38].

Planning in incomplete domain models is similar to the concept of *Classical Planning*, except that some actions are not completely specified and there are annotations specifying *possible preconditions* and *effects* of some actions in the domain definition [23, 96, 61, 60]. To deal with incomplete information in domain models, Garland and Lesh [23] developed the first planning approach in the literature that allows annotations about incompleteness in the domain definition. After, in [96], Weber and Bryce developed a set of approaches to planning and acting in incomplete domain models, allowing annotations with regard to the incompleteness of actions in the domain model. Most recently, Nguyen et al. [61, 60] use the same formalism from [96] to develop more modern approaches to planning in incomplete domain models. In this thesis, we follow the formalism of incomplete domain models from [61, 60] and use incomplete STRIPS domain models for modeling incomplete domains.

2.2.1   Incomplete STRIPS Domain Models

To represent incomplete domain models, we define an incomplete STRIPS domain model by following the formalism from [61, 60], defined as $\widetilde{\mathcal{D}} = \langle \mathcal{R}, \widetilde{\mathcal{O}} \rangle$. Here, $\widetilde{\mathcal{O}}$ contains the definition of incomplete operators comprised of a six-tuple $\widetilde{op} = \langle pre(\widetilde{op}), \widetilde{pre}(\widetilde{op}), eff^+(\widetilde{op}), eff^-(\widetilde{op}),$ $\widetilde{eff}^+(\widetilde{op}), \widetilde{eff}^-(\widetilde{op}) \rangle$, where: $pre(\widetilde{op})$ and $eff(\widetilde{op})$ have the same semantics as in the STRIPS domain models; and *possible* preconditions $\widetilde{pre}(\widetilde{op}) \subseteq \mathcal{R}$ that *might* be required as preconditions, as well as $\widetilde{eff}^+(\widetilde{op}) \subseteq \mathcal{R}$ and $\widetilde{eff}^-(\widetilde{op}) \subseteq \mathcal{R}$ that *might* be generated as *possible* effects either as add or delete effects. An incomplete domain $\widetilde{\mathcal{D}}$ has a *completion set* $\langle\langle \widetilde{\mathcal{D}} \rangle\rangle$ comprising all possible domain models derivable from an incomplete one. Namely, the number of all possible domain models is $\langle\langle \widetilde{\mathcal{D}} \rangle\rangle = 2^K$, where $K = \sum_{\widetilde{op} \in \widetilde{\mathcal{O}}}(|\widetilde{pre}(\widetilde{op})| + |\widetilde{eff}^+(\widetilde{op})| + |\widetilde{eff}^-(\widetilde{op})|)$. An incomplete STRIPS planning problem derived from an incomplete STRIPS domain $\widetilde{\mathcal{D}}$ and a set of typed objects Z is defined as $\widetilde{\mathcal{P}} = \langle \mathcal{F}, \widetilde{\mathcal{A}}, \mathcal{I}, G \rangle$, where: $\mathcal{F}$ is the set of facts (instantiated predicates from Z), $\widetilde{\mathcal{A}}$ is the set of incomplete instantiated actions from $\widetilde{\mathcal{O}}$ with objects from Z, $\mathcal{I} \subseteq \mathcal{F}$ is the initial state, and $G \subseteq \mathcal{F}$ is the goal state.

Most approaches for planning in incomplete domains [96, 61, 60] assume that plans succeed under the *most optimistic* conditions, which are:

- Possible preconditions $\widetilde{pre}$ do not need to be satisfied in a state S;

- Possible add effects $\widetilde{eff}^+$ are always assumed to occur in the resulting state S';

- Delete effects $\widetilde{eff}^-$ are ignored in the resulting state S'.

Therefore, formally, an incomplete action $\widetilde{a}$ instantiated from an incomplete operator $\widetilde{op}$ is applicable to a state S iff $S \models pre(\widetilde{a})$ and results in a new state S' such that $S' := (S/eff^-(a)) \cup (\widetilde{eff}^+(\widetilde{a}) \cup eff^+(a))$. Thus, a valid plan π that achieves a goal G from $\mathcal{I}$ in an incomplete planning problem $\widetilde{\mathcal{P}}$ is a sequence of actions that induces an *optimistic* sequence of states. Example 2.1 from Weber and Bryce [96] illustrates an abstract incomplete planning problem and a valid plan for it.

Example 2.1. *Consider the following incomplete planning problem $\widetilde{\mathcal{P}}$, where:*

- $\mathcal{F} = \{p, q, r, g\}$;

- $\widetilde{\mathcal{A}} = \{\widetilde{a}, \widetilde{b}, \widetilde{c}\}$, *where:*

$$- \; pre(\widetilde{a}) = \{p,q\}, \widetilde{pre}(\widetilde{a}) = \{r\}, \widetilde{\mathit{eff}}^{+}(\widetilde{a}) = \{r\}, \widetilde{\mathit{eff}}^{-}(\widetilde{a}) = \{p\}$$

$$- \; pre(\widetilde{b}) = \{p\}, \mathit{eff}^{+}(\widetilde{b}) = \{r\}, \mathit{eff}^{-}(\widetilde{b}) = \{p\}, \widetilde{\mathit{eff}}^{-}(\widetilde{b}) = \{q\}$$

$$- \; pre(\widetilde{c}) = \{r\}, \widetilde{pre}(\widetilde{c}) = \{q\}, \mathit{eff}^{+}(\widetilde{c}) = \{g\}$$

- $\mathcal{I} = \{p,q\}$; and

- $G = \{g\}$.

The $[\widetilde{a}, \widetilde{b}, \widetilde{c}]$ sequence of actions is a valid plan to achieve goal state $\{g\}$ from the initial state $\{p,q\}$. It corresponds to the optimistic state sequence: $s_0 = \{p,q\}, s_1 = \{p,q,r\}, s_2 = \{q,r\}, s_3 = \{q,r,g\}$. The number of completions for this example is $|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle| = 2^5$ (2 possible preconditions and 3 possible effects, i.e., 1 possible add effect and 2 possible delete effects).

2.3 Landmarks

In the *Planning* literature [35, 94, 80], *landmarks* are defined as *necessary* properties (alternatively, actions) that must be true (alternatively, executed) at some point in every valid plan (see Definition 5) to achieve a particular goal from an initial state, being often partially ordered following the sequence in which they must be achieved. Hoffman et al. [35] define *fact landmarks* (Definitions 6) as follows:

Definition 6 (Fact Landmark). *Given a planning instance* $\Pi = \langle \Xi, \mathcal{I}, G \rangle$, *a formula L is a landmark in Π iff L is true at some point along all valid plans that achieve G from $\mathcal{I}$. In other words, a landmark is a type of formula (e.g., conjunctive formula or disjunctive formula) over a set of facts that must be satisfied (or achieved) at some point along all valid plan executions.*

Vidal and Geffner [94] define *action landmarks* (Definition 7) as necessary actions that must be executed at some point along all valid plans that achieve a goal state G from an initial state $\mathcal{I}$. In this thesis, we do not explicitly use the concept of action landmarks, but rather use fact landmarks to build our planning–based approaches to goal recognition.

Definition 7 (Action Landmark). *Given a planning instance* $\Pi = \langle \Xi, \mathcal{I}, G \rangle$, *an action A is a landmark in Π iff A is a necessary action that must be executed at some point along all valid plans that achieve G from $\mathcal{I}$.*

From the concept of fact landmarks, Hoffmann et al. [35] introduce two types of landmarks as formulas: *conjunctive* and *disjunctive landmarks*. A *conjunctive landmark* is a
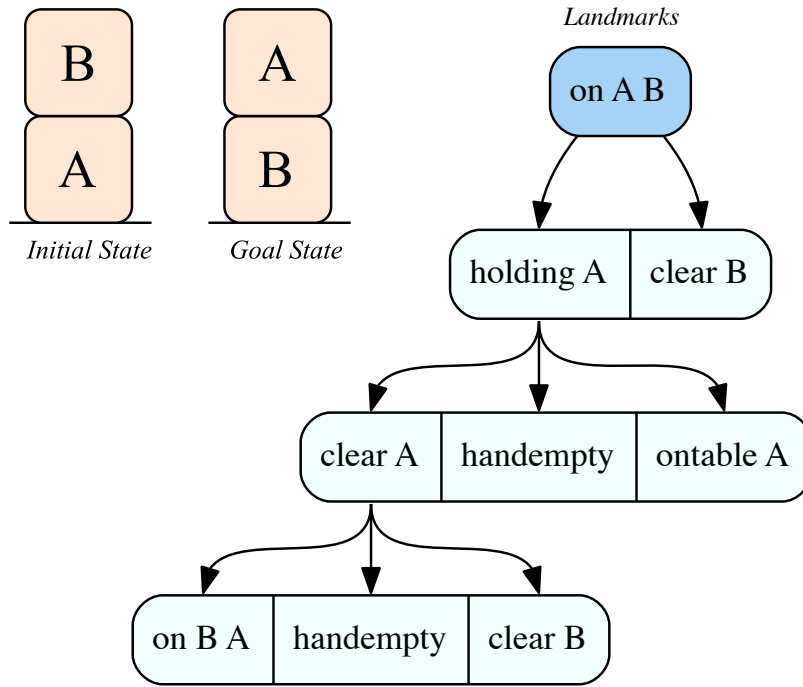
Figure 2.1: Ordered Landmarks for a BLOCKS-WORLD problem instance. Connected boxes represent conjunctive landmarks.

set of facts that must be true *together* at some point in every valid plan to achieve a goal. A *disjunctive landmark* is a set of facts such that *at least one* of the facts must be true at some point in every valid plan to achieve a goal. Figure 2.1 shows an example that illustrates a set of landmarks for a BLOCK-WORLD[1] problem instance. This example shows a set of conjunctive ordered landmarks (connected boxes) that must be true to achieve the goal state (`on A B`). For instance, to achieve the fact landmark (`on A B`) which is also the goal state, the conjunctive landmark (`and (holding A) (clear B)`) must be true immediately before, and so on, as shown in Figure 2.1.

Whereas in *Planning* the concept of landmarks is used to build heuristics [80] and planning algorithms [81], in this thesis, much like our previous work [65, 70], we exploit the concept of landmarks to reason about agents' plan execution, and attempt to recognize the goals that such agent aims to achieve. Intuitively, we use landmarks as waypoints (or stepping stones) in order to monitor what an observed agent cannot avoid to achieve its goals. In Chapter 3, we introduce new notions of landmarks and develop an algorithm for extracting landmarks over incomplete domain models.

---

[1]BLOCKS-WORLD is a *Classical Planning* domain where a set of stackable blocks must be re-assembled on a table [28, Chapter 2, Page 50].

2.4   Goal and Plan Recognition as Planning

Goal Recognition is the task of recognizing which goal an observed agent aims to achieve by observing its interactions in an environment [88, Chapter 1, Page 3]. *Plan Recognition* can be seen as a superset of goal recognition (Figure 2.2), namely, it is the task of recognizing which plan is being executed by an observed agent by observing its interactions in an environment [88, Chapter 3, Page 57]. In *Goal and Plan Recognition*, such observed interactions (i.e., observations) are used as available evidence to recognize goals and plans. Observed interactions can be observed events performed by an agent in an environment, as well as actions (e.g., a simple movement, cook, drive), and changing properties in an environment (e.g., at home, at work, resting). Approaches to goal and plan recognition are characterized according to the role that the observed agent performs during the recognition process in an environment [4, 88], as follows.

- **Intended Recognition** is the recognition process in which the observed agent is aware of the process of recognition. Therefore, in this kind of recognition process the observed agent usually cooperates with the process by notifying the recognizer about its interactions in the environment;

- **Keyhole Recognition**[2] is defined as the recognition process in which the observed agent is unaware of the process of recognition, namely, the interactions performed by the observed agent are partially observable inputs to the recognition process; and

- **Obstructed Recognition** is the recognition in which the observed agent is aware of the process of recognition and obstructs purposely the process. In other words, the agent intentionally does not cooperate with the recognition process.

To recognize goals and plans from agents' observations, we need a model that describes the agents' behavior and the environment, more specifically, a model that describes how observed agents act to achieve their goals in an environment. In the literature, the most commonly used types of models for goal and plan recognition are: *plan–libraries* and *planning domain definition*. A *plan–library* can be seen as a domain-specific model that defines a predefined and static set of plans to achieve a set of goals, i.e., a know-how for achieving goals in

---

[2]We note that the recognition approaches proposed in this thesis are limited to keyhole recognition, in which we make the assumption that the observed agents are either not aware that they are being observed or do not care about the recognition process.
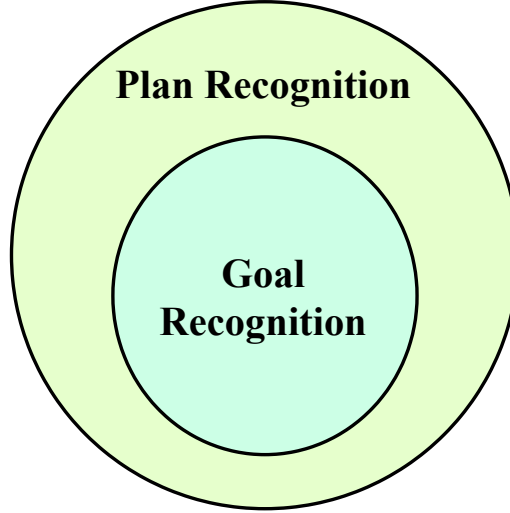
Figure 2.2: Plan Recognition as a superset of Goal Recognition [63].

a particular domain. While the use of plan–libraries has been proved to be computationally efficient [6, 7, 27, 55, 52], the process of modeling a plan–library requires substantial domain knowledge because the space of all possible plans for achieving goals must be encoded beforehand for every agent. Alternatively, goal and plan recognition approaches that use a model based on a *planning domain definition* are called as *Plan Recognition as Planning* [77, 78]. The use of *planning domain definition* for goal and plan recognition relaxes the need of modeling all plans to achieve a set of goals, by using only a planning domain-theory to describe facts and actions of the environment as domain knowledge. *Plan Recognition as Planning* brings the process of goal and plan recognition closer to planning algorithms, allowing the use of automated planners [77, 78, 87] and planning concepts [63, 19, 65, 70] for recognizing goals and plans. Ramírez and Geffner [77] claim that plan recognition can be seen as planning in reverse, namely, in plan recognition we search for goals and plans that better explain the observations, in planning we search for a sequence of actions (i.e., a plan) to achieve a particular goal.

In this thesis, we define the problem of goal recognition as planning by following the formalism proposed by Ramírez and Geffner in [77, 78], as formalized Definition 8.

Definition 8 (**Goal Recognition Problem as Planning**). *A goal recognition problem over a planning domain definition is a four-tuple* $\mathcal{T} = \langle \Xi, \mathcal{I}, \mathcal{G}, Obs \rangle$*, where:*

- $\Xi = \langle \mathcal{F}, \mathcal{A} \rangle$ *is a planning domain definition;*

- $\mathcal{I}$ *is the initial state;*

- $\mathcal{G}$ *is the set of hypothetical candidate goals, which include a correct hidden goal* G^* *(i.e.,* $G^* \in \mathcal{G}$*); and*

- $Obs = \langle o_1, o_2, ..., o_n \rangle$ *is an observation sequence of executed actions, with each observation* $o_i \in \mathcal{A}$*.*

We now formally define an observation sequence (Definition 9) by following the formalism proposed by Ramírez and Geffner in [77, 78], which defines an observation sequence as an action sequence.

Definition 9 (**Observation Sequence**). *An observation sequence* $Obs = \langle o_1, o_2, ..., o_n \rangle$ *is said to be satisfied by a plan* $\pi = \langle a_1, a_2, ..., a_m \rangle$*, if there is a monotonic function* f *that maps the observation indices* $j = 1, ..., n$ *into action indices* $i = 1, ..., n$*, such that* $a_{f(j)} = o_j$*.*

Thus, the ideal solution for a goal recognition problem as planning is finding the single correct hidden goal $G^* \in \mathcal{G}$ that the observation sequence *Obs* of a plan execution achieves. Most approaches to goal and plan recognition return either a probability distribution over the goals [77, 78, 87, 19], or a score associated to the set of hypothetical candidate goals [65, 70]. In Chapters 3 and 4, we explain how our approaches to goal recognition compute scores and probability distributions for estimating the correct hidden goal from the observation sequence. As an example of how the goal recognition process works, consider the Example 2.2, as follows.

Example 2.2. *To exemplify the goal recognition process, let us consider the* BLOCKS-WORLD *example in* Figure 2.3*. The initial state represents an initial configuration of stackable blocks, while the set of candidate goals is composed by the following stacked "words":* RED, BED, *and* SAD*. Consider an observation sequence for a hidden goal* RED *consisting of the following action sequence:* [(unstack D B), (putdown D), (unstack E A), (stack E D), (pickup R), (stack R E)]*. By following the full observation sequence, we can easily infer that the hidden goal is indeed* RED*. However, if the we cannot observe the action* (stack R E)*, it is not trivial to infer that* RED *is indeed the goal the observation sequence aims to achieve. Thus, we could infer that more than one candidate goal could be pursuit by the observations.*

The observation sequence in Example 2.2 represents a full observation sequence, i.e., all actions of an agent's plan are observed. In a partial observation sequence, we observe only a sub-sequence of actions of a plan that achieves a particular goal because some actions are missing or obfuscated. A noisy observation sequence contains one or more actions (or a set
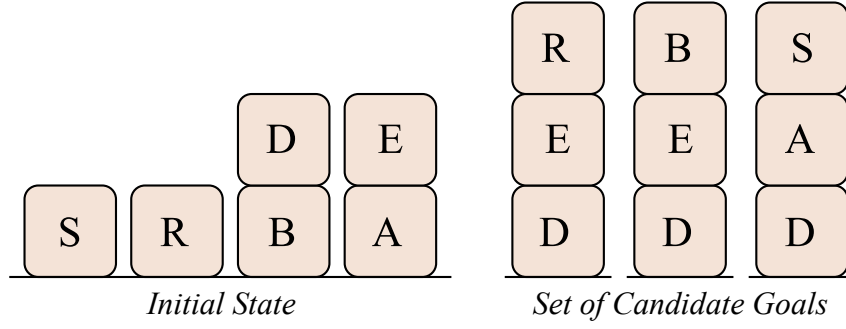
Figure 2.3: Goal Recognition example for BLOCKS-WORLD domain.

of facts) that might not be part of a plan that achieves a particular goal, e.g., when a sensor fails and generates abnormal or spurious readings [87]. In this thesis, we deal with partial and full observation sequences during the goal recognition process. We formally define a missing observation in Definition 10. Example 2.3 shows an example of a partial observation sequence containing misses observed actions.

Definition 10 (Missing Observation Sequence). *Let $\Pi = \langle \Xi, \mathcal{I}, G \rangle$ be a planning instance, π a valid plan that achieves G from $\mathcal{I}$, and Obs an observation sequence that satisfies π. An observation sequence Obs misses observations (is a partial observation sequence) with respect to the plan π that achieves the goal G from $\mathcal{I}$, if at least one of the observations $o \in Obs$ is part of a strict subset of ordered actions π.*

Example 2.3. *Let us consider that a valid plan to achieve a goal G is $\pi = [a, b, c, d, e]$. Consider the following observation sequences O_{m1}, O_{m2}, and O_{m3}:*

- *$O_{m1} = [a, d]$;*

- *$O_{m2} = [b, e]$; and*

- *$O_{m3} = [d, a, c]$*

Observation sequences O_{m1} and O_{m2} satisfy Definition 10, and therefore, they are partial observation sequences and contain missing observed actions. O_{m3} is not a partial observation sequence because it does not satisfy Definition 10 as the observation sequence $[d, a, c]$ is not a strict subset of ordered actions of the plan π.

2.5   Optimal Control

In the fields of *Engineering* and *Mathematics*, *Control Theory* is a subfield of study that deals with the behavior of dynamical systems [10]. A *dynamical system* can be seen as a mathematical model that aims to describe the behavior of a system. In a sense, a dynamical system is a system that describes mathematically the changes that occur over time in a physical system with geometrical space. This geometrical space is a set of possible states of the system, whereas the dynamics can be formalized as a function that transforms a state into another state. Thus, the aiming of the *Control Theory* is to control a dynamical system such that its output follows a desired value, which may be a fixed or changing value.

To model the range of possible agent behavior in *Control*, we use one of the branches of *Control Theory*, namely, *Optimal Control*. The objective of *Optimal Control* is either to minimize or maximize a particular measure regarding the behavior of a dynamical system over time [10, 89]. In this thesis, we model agent behavior using *Finite-Horizon Optimal Control* problems, denoted as FHOC.

2.5.1   Finite Horizon Optimal Control Problems

We formalize FHOC problems by following the terminology proposed by Bertsekas in [10], incorporating and combining some elements typically used by the literature on *Control* [13] and *Planning* [24] to account for constraints and goals[3]. Transitions between states are described by a *stationary*, *discrete–time* dynamical system

$$x_{k+1} = f(x_k, u_k, w_k) \tag{2.2}$$

where for each time point $k \in [0, N]$, x_k is the *state*, u_k is the *control input* and w_k is a *random variable* with a probability distribution that does not depend on past w_j, $j < k$. For now, we make no further assumptions on the specific way states, inputs and perturbations interact. States x_k, controls u_k, and disturbances w_k are required to be part of spaces $S \subset \mathbb{R}^d$, $C \subset \mathbb{R}^p$, and $D \subseteq \mathbb{R}^{d+p}$. Controls u_k are further required to belong to the set $U(x_k) \subset C$, for each

---

[3]Referred to as *target regions* in *Control Theory*.

state x_k and time step k. We note that the latter accounts for both the notion of *preconditions* and *bounds* on inputs. Observed agents seek to transform initial states x_0 into states x_N with specific properties. These properties are given as logical formulas over the components of states x_k, and the set of states $S_G \subseteq S$ are those where the desired property G, or goal, holds. The preferences of observed agents to pursue specific trajectories are accounted for with cost functions of the form

$$J(x_0) = \mathbb{E}\{g(x_N) + \sum_{k=0}^{N-1} g(x_k, u_k, w_k)\} \tag{2.3}$$

$g(x_N)$ is the *terminal cost*, $g(x_k, u_k, w_k)$ is the *stage cost*, and $\mathbb{E}$ is the expectation operator with respect to the random variable w_k. Thus, we define FHOC problems as an optimization problem whose solutions describe the range of possible optimal behaviors of observed agents

$$\min_{\pi \in \Pi}\left\{J_\pi\left(x_0\right)\right\} \tag{2.4}$$

subject to

$$u_k = \mu_k(x_k) \tag{2.5}$$

$$x_{k+1} = f(x_k, u_k, w_k) \tag{2.6}$$

$$u_k \in U(x_k), \ x_k \in S, \ x_N \in S_G \tag{2.7}$$

where $\mathcal{I}$, the *initial state*, is an arbitrary element of the set of states S. Solutions to Equations 2.4–2.7 are *policies* π

$$\pi = \{\mu_0, \mu_1, \ldots, \mu_k, \ldots, \mu_{N-1}\} \tag{2.8}$$

and μ_k is a function mapping states x_k into controls $u_k \in C$. When π is such that $\mu_i = \mu_j$ for every i, $j \in [0, N]$, we say π is *stationary*. We note the abuse of notation, and use the π symbol to represent both *plans* and *policies* throughout the text and clarify them when this is not obvious from context. We also note that terminal constraints $x_N \in S_G$ can be dropped, replacing them by terms in $g(x_k, u_k, w_k)$ that encode some measure of distance to S_G. Costs $g(x_N)$ are typically set to 0 when terminal constraints are enforced, yet this is a convention, and establishing preferences for specific states in S_G over others is perfectly possible.

2.5.2 Actual and Nominal Control Models

System Identification is the task of building and approximating mathematical models of dynamical systems from past collected data [47] (e.g., observed state transitions) and prior system knowledge. Existing work on the *Control Theory* literature [58, 45] usually refers to approximate models as *nominal models*, whereas correct true (idealized) models are referred as *actual models*. We formally define *actual* and *nominal models* as follows, in Definitions 11 and 12, respectively.

Definition 11 (Actual Model). *An actual model is a model in which the transition function* $x_{k+1} = f(x_k, u_k, w_k)$ *is known and well defined.*

Definition 12 (Nominal Model). *A nominal model is a model in which the transition function* $\hat{x}_{k+1} = \hat{f}(x_k, u_k, w_k)$ *is approximate and acquired based on observed data.*

In this thesis, we study the implications of using *nominal models* for goal recognition over continuous actions and state spaces. In Chapter 4, we show how we acquire *nominal models* from past observed state transitions by leveraging existing work that combines *Deep Learning* and *Planning* [85, 97].

3.   GOAL RECOGNITION OVER INCOMPLETE DOMAIN MODELS

Traditional work on *Goal and Plan Recognition as Planning* relies on complete and correct domain models, relaxing the need to deal with incomplete domain information in relation to methods based on plan–libraries. In this chapter, we present novel heuristic approaches to goal recognition that explicitly deal with *incomplete domain information over discrete models*. In Section 3.1, we introduce the problem of goal recognition over *incomplete domain models*. After, in Section 3.2, we present new notions of *landmarks* for *incomplete domain models*, and how we modify an algorithm from the literature to extract such landmarks from incomplete domain information. Then, in Section 3.3, we build goal recognition heuristics based on these new notions of landmarks over *incomplete domain models*. Finally, in Section 3.4, we show empirically that our heuristic approaches effectively deal with the incomplete part of the domain model to substantially improve recognition accuracy against a baseline.

3.1   Problem Formulation

Most planning–based approaches to goal and plan recognition assume that a complete domain model is available to perform the recognition process [77, 78, 40, 19, 87, 65, 70], relying on conventional models in which the domain information is considered to be *known*, *correct*, and *well-defined*. In this thesis, we propose the problem of goal recognition over *incomplete domain models*. Here, we assume that the *observer* (goal recognizer) uses an *incomplete domain model* to perform the recognition task, while the *observed agent* is planning and acting in the environment with a *complete* domain model. Besides that, like the planning approaches over incomplete domain models [96, 60] (recall from Section 2.2), we consider that the goal recognizer reasons about possible plans with incomplete actions (observations) by assuming that they succeed under the *most optimistic* conditions, i.e., ignoring all *possible* preconditions and delete effects (known and possible), and all effects are assumed to occur (known and possible).

To account for such incompleteness in the domain model, the incomplete domain model available to the observer contains annotations specifying *possible preconditions* and *effects* of some actions in the incomplete domain definition, much like the incomplete domain models

from previous planning approaches [96, 60]. Figure 3.1 shows an overview of how we define the problem of goal recognition over *incomplete domain models* from the perspective of both the observer and the observed agent.
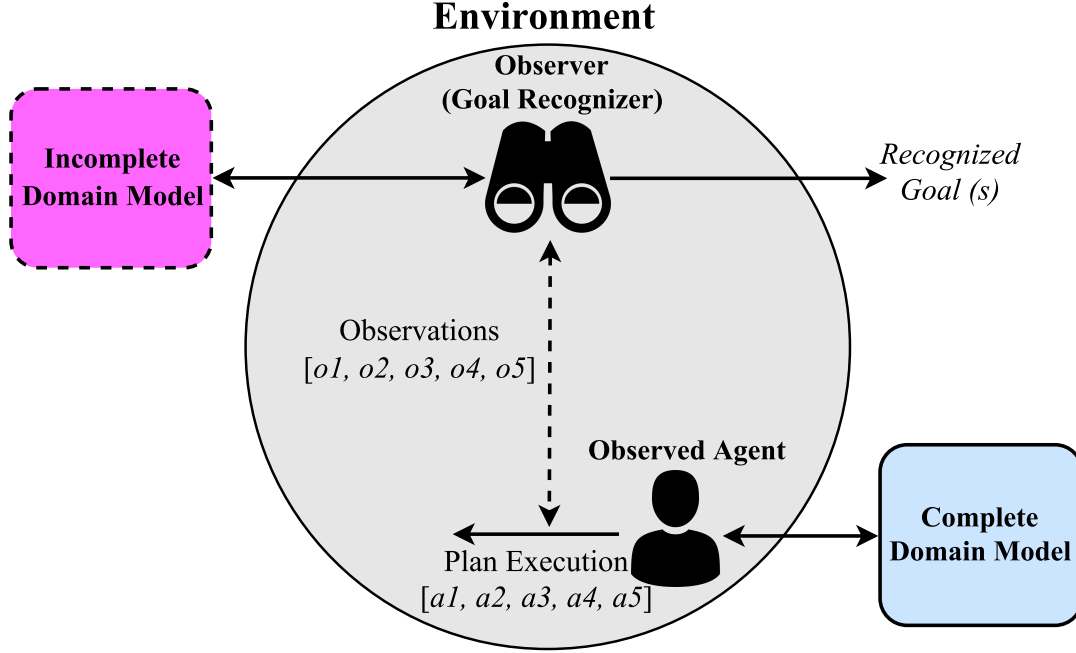
Figure 3.1: Problem overview to goal recognition in incomplete domain models.

By combining the various notions of planning in incomplete domains [96, 60] and observation sequence (Definition 9), we now formally define the *goal recognition problem* over an *incomplete planning domain* in Definition 13, following the formalism of Ramírez and Geffner [77, 78] (Definition 8).

Definition 13 (**Goal Recognition Problem over an Incomplete Domain Model**). *A goal recognition problem over an incomplete domain model is a quintuple $\widetilde{\mathcal{T}} = \langle \widetilde{\mathcal{D}}, Z, \mathcal{I}, \mathcal{G}, Obs \rangle$, where:*

- *$\widetilde{\mathcal{D}} = \langle \mathcal{R}, \widetilde{\mathcal{O}} \rangle$ is an incomplete domain model (with possible preconditions and effects). Z is the set of typed objects in the environment, in which $\mathcal{F}$ is the set instantiated predicates (i.e., a finite set of facts) from $\mathcal{R}$ with objects from Z, and $\widetilde{\mathcal{A}}$ is the set of incomplete instantiated actions from $\widetilde{\mathcal{O}}$ with objects from Z;*

- *$\mathcal{I} \in \mathcal{F}$ is the initial state;*

- *$\mathcal{G}$ is the set of hypothetical candidate goals, which include a correct hidden goal G^* (i.e., $G^* \in \mathcal{G}$); and*

- $Obs = \langle o_1, o_2, ..., o_n \rangle$ *is an observation sequence of executed actions* $o_i \in \widetilde{\mathcal{A}}$. *Obs corresponds to a sequence of actions (i.e., a plan) that achieves the correct hidden goal* G^* *from the initial state* $\mathcal{I}$ *over a single complete model* $\mathcal{D}^* \in \langle\langle \widetilde{\mathcal{D}} \rangle\rangle$ *that is known to the observed agent, but not to the recognizer.*

Ideally, a solution for a goal recognition problem in incomplete domain models $\widetilde{\mathcal{T}}$ is to find the single correct hidden goal $G^* \in \mathcal{G}$ that the observation sequence *Obs* of a plan execution achieves. As most goal recognition approaches, observations consist of the action signatures of the underlying plan[1], more specifically, we observe incomplete actions with possible precondition and effects, in which some of the preconditions might be required and some effects might change the environment. A full (or complete) observation sequence contains all of the action signatures of the plan executed by the observed agent, whereas a partial observation sequence contains only a sub-sequence of actions of a plan, and thus misses some of the actions actually executed in the environment (Definition 10). In summary, the problem we address in this thesis is *keyhole goal recognition* under partial observability [88, Chapter 1].

3.2 Extracting Landmarks in Incomplete Domain Models

Recall from Section 2.3 that landmarks are necessary conditions to achieve a goal from an initial state in a given planning problem (Definitions 6 and 7). In the *Planning* literature, most landmark–based heuristics extract landmarks from complete and correct domain models. In this thesis, we introduce new notions of fact landmarks in *incomplete domain models*, and extend the landmark extraction algorithm proposed by Hoffmann et al. in [35] to extract landmarks from incomplete STRIPS domain models.

To represent landmarks and their ordering, the algorithm of Hoffmann et al. [35] uses a tree in which nodes represent landmarks and edges represent necessary prerequisites between landmarks. Each node in the tree represents a conjunction of facts that must be true simultaneously at some point during plan execution, and the root node is a landmark representing the goal state. For extracting landmarks, this landmark extraction algorithm [35] uses a Relaxed Planning Graph (RPG) [14], which is a leveled graph that ignores the delete-list effects of all actions, thus containing no mutex relations [34], as formalized in Definition 14.

---

[1]Our approaches are not limited to using just actions as observations, and it can also deal with logical facts as observations, much like the approach from [87].

Algorithm 1 shows a pseudo-code to build a full RPG structure from a STRIPS planning problem $\mathcal{P}$.

Definition 14 (Relaxed Planning Graph). *An RPG is a leveled graph structure that ignores the delete-list effects of all actions, and in this way, there are no mutex relation in this graph structure. Thus, as a leveled graph, the graph levels are structured as follows: $\mathcal{F}_0$, $\mathcal{A}_0$, $\mathcal{F}_1$, $\mathcal{A}_1$, ..., $\mathcal{F}_{m-1}$, $\mathcal{A}_{m-1}$, $\mathcal{F}_m$ of fact sets $\mathcal{F}_i$ (fact levels) and action sets $\mathcal{A}_i$ (action levels). Fact level $\mathcal{F}_0$ contains the facts that are true in the initial state, the action level $\mathcal{A}_0$ contains those actions whose preconditions are reached from $\mathcal{F}_0$, subsequently, $\mathcal{F}_1$ contains $\mathcal{F}_0$ plus the add effects of the actions in $\mathcal{A}_0$. The graph is built until reaching the goal state in the last fact level, or when the building process fails, if at some point before reaching the goals no new facts are inserted in the graph. Thus, the RPG is composed of $\mathcal{F}_i \subseteq \mathcal{F}_{i+1}$ and $\mathcal{A}_i \subseteq \mathcal{A}_{i+1}$ for all i.*

Algorithm 1 Build a Relaxed Planning Graph (RPG).

Input: $\mathcal{P} = \langle \mathcal{F}, \mathcal{A}, \mathcal{I}, G \rangle$ STRIPS *planning problem, in which $\mathcal{F}$ is a finite set of facts and $\mathcal{A}$ is a finite set of actions, $\mathcal{I}$ is the initial state, and G is the goal state.*
Output: RPG *relaxed planning graph.*

1: **function** BUILDFULLRPG($\mathcal{F}, \mathcal{A}, \mathcal{I}, G$)
2:     $i := 0$
3:     RPG.$\mathcal{F}_0 := \mathcal{I}$ ▷ *Initialize the* RPG *with facts of the initial state.*
4:     **while** $G \nsubseteq$ RPG.$\mathcal{F}_i$ **do**
5:         RPG.$\mathcal{A}_i :=$ all action $a \in \mathcal{A}$ such that $pre(a) \in$ RPG.$\mathcal{F}_i$
6:         RPG.$\mathcal{F}_{i+1} :=$ RPG.$\mathcal{F}_i \cup (eff^+(a), \forall a \in$ RPG.$\mathcal{A}_i)$
7:         **if** RPG.$\mathcal{F}_{i+1} \equiv$ RPG.$\mathcal{F}_i$ **then**
8:             **return** FAILURE TO BUILD RPG ▷ *The algorithm fails whether at some point before reaching the facts of the goal no new fact level is added in the graph.*
9:         $i := i + 1$
10:    **return** RPG

After building the RPG, the algorithm of Hoffmann et al. [35] extracts a set of *landmark candidates* by back-chaining from the RPG level in which all facts of the goal state G are possible, and, for each sub-goal (fact) g in G, it checks which facts must be true until the first level of the RPG. For example, if a fact LB is a landmark and all actions that achieve LB share LA as a precondition, then LA is a landmark candidate. To confirm that a landmark candidate is indeed a necessary fact to achieve the goal state G, and thus an actual landmark, the algorithm builds a new RPG removing all actions that achieve the landmark candidate and checks the solvability over this modified problem. If the modified problem is *unsolvable*, then such landmark candidate is a necessary fact to achieve G, and therefore, a fact landmark. This means that the actions that achieve the landmark candidate are necessary to solve the

original planning problem. According to Blum and Furst [12], deciding the solvability of a relaxed planning problem using an RPG structure can be done in polynomial time.

3.2.1 Optimistic Relaxed Planning Graph

In order to extract landmarks in *incomplete domain models*, we develop and formalize an Optimistic Relaxed Planning Graph (ORPG). An ORPG is leveled graph that deals with incomplete domain models by assuming the most *optimistic* conditions. Thus, besides ignoring the delete-effects of all actions, this graph also ignores possible preconditions and possible delete-effects, whereas we use all possible and known add effects to build the optimistic graph. We formally define an ORPG in Definition 15.

Definition 15 (Optimistic Relaxed Planning Graph). *An ORPG is a leveled graph and similar to an RPG structure, however, it considers the optimistic assumption, in which possible preconditions do not need to be satisfied in a state, possible add effects are always assumed to occur, and delete effects are always ignored. As a leveled graph, the graph levels in the ORPG are structured as follows: $\widetilde{\mathcal{F}}_0, \widetilde{\mathcal{A}}_0, \widetilde{\mathcal{F}}_1, \widetilde{\mathcal{A}}_1, ..., \widetilde{\mathcal{F}}_{m-1}, \widetilde{\mathcal{A}}_{m-1}, \widetilde{\mathcal{F}}_m$ of fact sets $\widetilde{\mathcal{F}}_i$ (fact levels) and action sets $\widetilde{\mathcal{A}}_i$ (incomplete action levels). Fact level $\widetilde{\mathcal{F}}_0$ contains the facts that are true in the initial state, while the incomplete action level $\widetilde{\mathcal{A}}_0$ contains those actions whose preconditions are reached from $\widetilde{\mathcal{F}}_0$ (possible preconditions are ignored). $\widetilde{\mathcal{F}}_1$ contains $\widetilde{\mathcal{F}}_0$, plus the facts in the add effects and possible add effects of the incomplete actions in $\widetilde{\mathcal{A}}_0$, and so on until reaching the goal state in the last fact level, or when the building process fails, if at some point before reaching the goals no new facts are inserted in the graph. Thus, the ORPG structure is composed of $\widetilde{\mathcal{F}}_i \subseteq \widetilde{\mathcal{F}}_{i+1}$ and $\widetilde{\mathcal{A}}_i \subseteq \widetilde{\mathcal{A}}_{i+1}$ for all i.*

In practice, to build an ORPG, we must modify four steps in the original RPG algorithm (formally described in Algorithm 1) resulting in a function called BUILDFULLORPG. Below we show the necessary modifications that must be done in Algorithm 1 to build a full ORPG for a given incomplete STRIPS planning problem.

1. The input must be an incomplete STRIPS planning problem $\widetilde{\mathcal{P}} = \langle \mathcal{F}, \widetilde{\mathcal{A}}, \mathcal{I}, G \rangle$;

2. Build (Line 3) and output an ORPG instead of an RPG;

3. In Line 5: create a new action level by selecting all incomplete instantiated action $\widetilde{a} \in \widetilde{\mathcal{A}}$ such that $pre(\widetilde{a}) \in \text{ORPG}.\widetilde{\mathcal{F}}_i$; and

4. In Line 6: create a new fact level with ORPG.$\widetilde{\mathcal{F}}_i \cup (\mathit{eff}^+(\tilde{a}) \cup \widetilde{\mathit{eff}}^+(\tilde{a}), \forall \tilde{a} \in$ ORPG.$\widetilde{\mathcal{A}}_i)$;

Replacing an RPG for an ORPG allows us to detect new types of landmarks over incomplete domain models, and thus, extract more landmarks than the original algorithm of Hoffmann et al. [35]. Next, we introduce new notions of landmarks and a novel landmark extraction algorithm that accounts for the incompleteness in incomplete STRIPS domain models.

3.2.2 Landmark Extraction Algorithm for Incomplete Domains

For extracting landmarks in incomplete domains models, we adapt the extraction algorithm developed by Hoffman et al. [35] to extract *definite* and *possible* landmarks from incomplete planning problems by building an ORPG instead of the original RPG and making other modifications. We formally define the notions of *definite* and *possible* landmarks in Definitions 16 and 17, respectively.

Definition 16 (Definite Landmark). *A definite landmark L_D is a fact (landmark) that is extracted from a known add effect $\mathit{eff}^+(a)$ of an achiever[2] a (action) in the ORPG.*

Definition 17 (Possible Landmark). *A possible landmark L_P is a fact (landmark) that is extracted from a possible add effect $\widetilde{\mathit{eff}}^+(a)$ of an achiever a (action) in the ORPG and is such that $L_P \cap L_D = \emptyset$.*

Unlike the algorithm proposed by Hoffman et al. [35] that besides extracting the set of landmarks also approximates the order that they must be achieved, our extraction algorithm only extracts the set of landmarks (i.e., a set of individual facts, not a set of formulas, as defined in Definition 6) for a given incomplete planning problem, avoiding the need to compute the order relation between the extracted landmarks. The landmark extraction algorithm for incomplete domain models we develop is formally described in Algorithm 2. Our landmark extraction algorithm takes as input an incomplete domain definition $\widetilde{\mathcal{D}} = \langle \mathcal{R}, \widetilde{\mathcal{O}} \rangle$, a set of typed objects Z, in which $\mathcal{F}$ is the set of facts instantiated from $\mathcal{R}$ and Z, $\widetilde{\mathcal{A}}$ is the set of incomplete actions instantiated from $\widetilde{\mathcal{O}}$ and Z, $\mathcal{I}$ is the initial state, and G is the goal state. Initially, the algorithm builds a full ORPG structure from an initial state to goal state (Line 2). If the goal state is not reachable in the ORPG, then landmarks are not extracted, whereas if

---

[2]An achiever is an action at the level before a candidate landmark in the RPG that can be used to achieve this candidate landmark.

the goal state is reachable, a set of landmark candidates C is initialized with the facts from the goal state (Line 7). Subsequently, the algorithm iterates over the set of landmark candidates in C (Line 8), and, for each landmark l, the algorithm identifies fact landmarks that must be true immediately before l. This iteration ends when the set of landmark candidates C is empty. Then, in Line 10, from landmark candidate l, the set of actions A' is extracted from the ORPG, comprising all actions in the ORPG at the action level immediately before l. In other words, A' represents those actions that achieve the facts in the level in which l is in the ORPG, these actions are called *achievers*. In Line 11, a for iteration filters the set A' for those actions such that can achieve l, i.e., actions that contain l in their (known or possible) effects, formally, $\forall \widetilde{a} \in A'$ such that $l \in (eff^+(\widetilde{a}) \cup \widetilde{eff}^+(\widetilde{a}))$. From the filtered actions A', our algorithm takes as new landmark candidates those facts that are in the preconditions of every action $\widetilde{a}$ in A' (Line 13). In Line 14, the algorithm checks if a landmark candidate f is indeed a landmark using a function called IsLandmark. This function evaluates whether a landmark candidate is a necessary condition to achieve a goal [35]. For example, consider an $ORPG'$ structure built from a planning problem $\mathcal{I}$ and G in which every action level in this $ORPG'$ does not contain actions that achieve the landmark candidate l'. Given this modified $ORPG'$ structure, we test the solvability of the planning problem $\mathcal{I}$ and G. So, if this problem is unsolvable, then the landmark candidate l' is indeed a landmark. More specifically, it means that the actions that achieve the fact l' must be part of the $ORPG'$ to solve this planning problem. Namely, the actions that contain the fact l' in their effects are necessary to solve the problem. Thus, in Line 15, the algorithm selects the facts which are indeed landmarks, and will be used extract other landmarks. The algorithm stores facts that must be true together (Line 16). In Lines 18 and 20, the algorithm stores fact landmarks according to their types, as formalized in Definitions 16 and 17. Finally, in Line 21, our algorithm returns the set of extracted *definite* and *possible* landmarks, respectively, $\mathcal{L}_G$ and $\widetilde{\mathcal{L}}_G$.

Based on the incomplete planning problem formalized in Example 2.1, we now exemplify how our adapted extraction algorithm extracts landmarks, as follows in Example 3.1.

Example 3.1. *Recalling that for Example 2.1, the set of facts is $\mathcal{F} = \{p, q, r, g\}$, the set of incomplete actions is $\widetilde{\mathcal{A}} = \{\widetilde{a}, \widetilde{b}, \widetilde{c}\}$, and the initial and goal states are $\mathcal{I} = \{p, q\}$ and $G = \{g\}$. Therefore, for this example, the set of definite and possible landmarks when using Algorithm 2 is $\{p, q, r, g\}$. The set of definite landmarks is $\{p, r, g\}$ (Light-Blue in Figure 3.2), and the set of possible landmarks is $\{q\}$ (Light-Yellow in Figure 3.2). The original landmark extraction algorithm proposed by Hoffmann et al. in [35] (without the most optimistic conditions), returns*

$\{p, r, g\}$ *as landmarks. The tradditional landmark extraction algorithm does not extract q as a fact landmark because it does not assume the most optimistic condition that possible add effects always occur, therefore, the action a was not considered as a possible achiever (action). Thus, by using our new extraction algorithm along with an ORPG instead an RPG we can extract not only definite landmarks but also possible landmarks, obtaining more landmarks than the original algorithm of Hoffmann et al. [35]. Figure 3.2 shows an ORPG for the incomplete planning problem illustrated in Example 2.1.*

Algorithm 2 Landmark Extraction Algorithm for Incomplete Domain Models.

Input: $\widetilde{\mathcal{D}} = \langle \mathcal{R}, \widetilde{\mathcal{O}} \rangle$ *is the incomplete domain definition, Z is a set of typed objects, where $\mathcal{F}$ is the set of facts instantiated from $\mathcal{R}$ and Z, $\widetilde{\mathcal{A}}$ is the set of incomplete actions instantiated from $\widetilde{\mathcal{O}}$ and Z, $\mathcal{I}$ is the initial state, and G is the goal state.*
Output: $\mathcal{L}_G$ *set of definite landmarks and $\widetilde{\mathcal{L}}_G$ set of possible landmarks.*

1: **function** EXTRACTLANDMARKS($\widetilde{\mathcal{D}}, Z, \mathcal{I}, G$)
2:     ORPG := BUILDFULLORPG($\mathcal{F}, \widetilde{\mathcal{A}}, \mathcal{I}, G$)
3:     **if** $G \nsubseteq$ ORPG.LASTFACTLEVEL **then**
4:         **return** FAILURE TO EXTRACT LANDMARKS
5:     $\mathcal{L}_G, \widetilde{\mathcal{L}}_G := \langle \ \rangle$ $\triangleright$ *Set of extracted (definite and possible) landmarks.*
6:     $\mathcal{L}_G := G$ $\triangleright$ *Trivially, the facts in the goal state are (definite) landmarks.*
7:     $C := G$ $\triangleright$ *Set of landmark candidates.*
8:     **while** $C \neq \emptyset$ **do**
9:         $l := C$.POP $\triangleright$ *Candidate landmark.*
10:         $A' := \text{ORPG}.\widetilde{\mathcal{A}}_{(\text{ORPG.LEVEL}(l)-1)}$ $\triangleright$ *All actions in the ORPG at the action level immediately before the candidate fact landmark l.*
11:         **for** each incomplete action $\widetilde{a}$ in A' such that $l \in (\textit{eff}^+(\widetilde{a}) \cup \widetilde{\textit{eff}}^+(\widetilde{a}))$ **do**
12:             $L_{Together} := \langle \ \rangle$ $\triangleright$ *Set of extracted fact landmarks that must be true together.*
13:             **for** each fact f in $pre(\widetilde{a})$ **do**
14:                 **if** ISLANDMARK($f, \mathcal{I}, G$) **then**
15:                     $C := C \cup f$
16:                     $L_{Together} := L_{Together} \cup f$
17:             **if** $L_{Together} \neq \emptyset$ and $l \in \textit{eff}^+(\widetilde{a})$ **then** $\triangleright$ *Check if l is a definite landmark.*
18:                 $\mathcal{L}_G := \mathcal{L}_G \cup L_{Together}$ $\triangleright$ *Add definite landmarks into $\mathcal{L}_G$.*
19:             **else if** $L_{Together} \neq \emptyset$ and $l \in \widetilde{\textit{eff}}^+(\widetilde{a})$ **then** $\triangleright$ *Check if l is a possible landmark.*
20:                 $\widetilde{\mathcal{L}}_G := \widetilde{\mathcal{L}}_G \cup L_{Together}$ $\triangleright$ *Add possible landmarks into $\widetilde{\mathcal{L}}_G$.*
21:     **return** $\mathcal{L}_G, \widetilde{\mathcal{L}}_G$

We note that in this thesis, we use this new landmark extraction algorithm to build heuristics to goal recognition over *incomplete domain models*. Although we use these new notations of landmarks for goal recognition, we argue that these new notions could be easily used to build heuristics for planning in incomplete domain models.
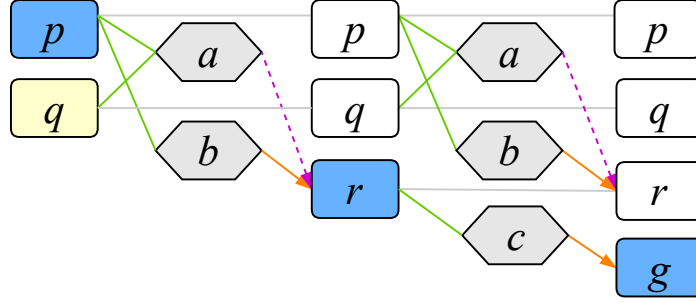
Figure 3.2: ORPG for the abstract incomplete planning problem presented in Example 2.1. Green arrows represent preconditions, Orange arrows represent add effects, and Purple dashed arrows represent possible add effects. Light-Blue boxes represent the set of *definite* landmarks and Light yellow boxes represent the set of *possible* landmarks. Grey hexagons represent actions.

3.3 Goal Recognition Heuristics over Incomplete Domain Models

We now develop novel goal recognition heuristics that rely on landmarks over *incomplete domain models*[3]. More specifically, we enhance recognition heuristics by exploiting the new notions of landmarks in *incomplete domain models*. Key to our enhanced heuristic approaches to goal recognition over *incomplete domain models* is collecting evidence of achieved landmarks during observations to recognize which goal is more consistent with the observations in a plan execution. Figure 3.3 illustrates the basic idea of our heuristic approaches to goal recognition over *incomplete domain models*. Basically, we start extracting landmarks for every goal in the set of candidate goals from the initial state. After that, we check which landmarks have been achieved in the observations for every candidate goal. Then, we compute a score for the goals based on the ratio of the achieved landmarks and the total amount of landmarks, and finally, we consider as the most likely goal the goals with the highest computed score. Because landmarks are necessary conditions on plans that achieve a goal, such heuristics can underpin the task of goal recognition when observing either optimal or sub-optimal behaviors. That is, since to achieve a goal from an initial state, an observed agent has to achieve all landmarks, regardless of whether the observed agent is executing an optimal or a sub-optimal plan.

To build our heuristic approaches to goal recognition over *incomplete domain models*, we exploit and combine the concepts of *definite* and *possible* landmarks, with that of *overlooked* landmarks (Definition 18).

---

[3]The heuristics we present in this section are enhanced heuristics based on recognition heuristics over *complete and correct domain models*. In APPENDIX A, we develop the original landmark–based heuristics for goal recognition over complete domains models, presenting practical examples and theoretical properties of such heuristics.
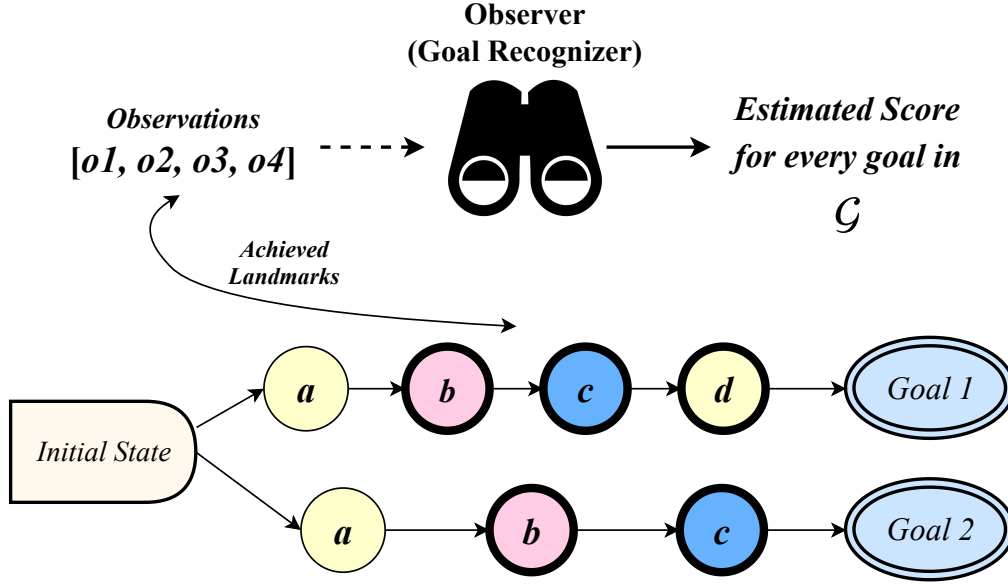
Figure 3.3: Heuristic Goal Recognition over Incomplete Domain Models.

Definition 18 (Overlooked Landmark). *An overlooked landmark L_O is an actual landmark, a necessary fact for all valid plans towards a goal from an initial state, that was not detected by approximate landmark extraction algorithms.*

Most landmark extraction algorithms extract only a sub-set of landmarks for a given planning problem [100, 35, 80, 44], and to overcome this issue and obtain more information to build heuristic approaches, we aim to extract *overlooked* landmarks by analyzing preconditions and effects in the observed actions of an observation sequence. Since we are dealing with incomplete domain models, and it is possible that some incomplete planning problems have few (or no) *definite* and/or *possible* landmarks, we extract *overlooked* landmarks from the evidence in the observations as we process them in order to enhance the set of landmarks useable by our goal recognition heuristics.

We extract *overlooked* landmarks *on the fly* (i.e., during the goal recognition process), and this *on the fly* extraction checks the facts in the preconditions and effects of the observations to extract landmarks that have not been extracted by our extraction algorithm developed in Section 3.2. Specifically, our *on the fly* extraction checks if the facts in the known preconditions and known possible add effects are not in the set of extracted *definite* and *possible* landmarks, and if they are not, we check if these facts are *overlooked* landmarks. In order to do that, we use the ISLANDMARK function that builds a new ORPG by removing actions that achieve a fact (i.e., a potentially *overlooked* landmark) and checks the solvability of this modified problem. If the modified problem is indeed unsolvable, then this fact is an *overlooked* landmark. We check

every candidate goal G in $\mathcal{G}$ using this function to extract additional (*overlooked*) landmarks. Example 3.2 illustrates how we extract *overlooked* landmarks *on the fly* for a given candidate goal and an observed action.

Example 3.2. *To exemplify how we extract overlooked landmarks during the recognition process, let us consider the goal state $\{g\}$ defined in Example 2.1 as a candidate goal G, for a recognition problem with initial state $\mathcal{I} = \{p, q\}$, and a sequence of observations $Obs = a$. Assume that a landmark extraction algorithm extracts $\mathcal{L} = \{p, q, g\}$ for initial state $\mathcal{I}$ and candidate goal G. Given the observed action a, we check if the facts in the preconditions and known/possible effects of a are in $\mathcal{L}$, and these facts are: $\{p, q, r\}$. Since r is not in $\mathcal{L}$, r can be an overlooked landmark. To check if r is an overlooked landmark, we build ORPG removing all actions that achieve r (i.e., actions a and b) and check whether G remains solvable in this ORPG. In this case, the goal atom g is unreachable (and G is unsolvable) because r is a necessary fact to achieve g. Figure 3.2 illustrates that g is unachievable without actions a and b, and consequently without r, because there is no action that adds this fact. Thus, r is an overlooked landmark that was not extracted by the extraction algorithm, but it was extracted on the fly from an observed action during plan execution.*

After presenting the basic idea of how our recognition heuristics work, next, we develop two goal recognition heuristics over incomplete domain models that rely on the notions of landmarks we introduced before. In Section 3.3.1, we develop a recognition heuristic that estimates *goal completion* by calculating the ratio between achieved landmarks and total amount of landmarks for a goal, and this heuristic is called *Enhanced Goal Completion Heuristic.* We then develop in Section 3.3.2 a recognition heuristic that uses the concept of *landmark uniqueness value*, representing the information value of the landmark for some specific candidate goal when compared to landmarks for all candidate goals, and this heuristic is called *Enhanced Uniqueness Heuristic.*

3.3.1 Enhanced Goal Completion Heuristic for Incomplete Domain Models

With the new notions of landmarks for incomplete domains in place, we now present a novel recognition heuristic for recognizing goals in incomplete domain models, called *Enhanced Goal Completion Heuristic*, denoted as $h_{\widetilde{GC}}$. We enhance the *Goal Completion Heuristic* (h_{gc}) presented in APPENDIX A to cope with incomplete domain models. Our enhanced heuristic

combines the notions of *definite*, *possible*, and *overlooked* landmarks, and estimates the correct intended goal in the set of candidate goals by calculating the ratio between achieved *definite* ($\mathcal{AL}_G$), *possible* ($\widetilde{\mathcal{AL}}_G$), and *overlooked* ($\mathcal{ANL}_G$) landmarks and the amount of *definite* ($\mathcal{L}_G$), *possible* ($\widetilde{\mathcal{L}}_G$), and *overlooked* ($\mathcal{NL}_G$) landmarks. This estimate, formalized in Equation 3.1, represents the percentage of achieved landmarks for a candidate goal from observations, more specifically, it represents the percentage of completion of a goal based on the achieved landmarks and the total number of extracted landmarks.

$$h_{\widetilde{GC}}(G) = \left(\frac{\mathcal{AL}_G + \widetilde{\mathcal{AL}}_G + \mathcal{ANL}_G}{\mathcal{L}_G + \widetilde{\mathcal{L}}_G + \mathcal{NL}_G} \right) \tag{3.1}$$

Even when a candidate goal has achieved all its landmarks during the recognition process for this heuristic, it is possible that such candidate goal is not the correct intended goal based on the observations. Such errors were possible in our approach for complete domains when recognizing goals from suboptimal plans. This limitation is exacerbated when we consider the set of *possible* landmarks ($\widetilde{\mathcal{L}}_G$) in the heuristic computation of $h_{\widetilde{GC}}$. This type of landmark is extracted from the incomplete part of the domain model, so they can be considered as "uncertain" landmarks, i.e., landmarks that might not be necessary conditions to achieve a goal from an initial state. In Section 3.4.5, we perform an ablation study and show the real impact of the new notations of landmarks on the recognition performance over our enhanced heuristics.

Example 3.3 illustrates how our enhanced goal completion heuristic works for recognizing goals over incomplete domains models.

Example 3.3. *To exemplify the recognition process using $h_{\widetilde{GC}}$, let us consider the goal recognition problem and the sets of landmarks presented in Figure 3.3. In this example, we have two candidate goals, Goal 1 (correct intended goal) and Goal 2, and the sets of landmarks for these goals are: Goal 1 = $\{a, b, c, d\}$ and Goal 2 = $\{a, b, c\}$. Blue circles represent definite landmarks (c), yellow circles represent possible landmarks (a,d), and pink circles represent overlooked landmarks (b). The achieved landmarks for the goals Goal 1 and Goal 2 are respectively: $\{b, c, d\}$ and $\{b, c\}$. By using our proposed enhanced goal completion heuristic $h_{\widetilde{GC}}$ to estimate which goal is the intended one in this example, we have the following heuristic estimation for Goal 1 and Goal 2:*

- $h_{\widetilde{GC}}(Goal\ 1) = \left(\frac{\mathcal{AL}_G + \widetilde{\mathcal{AL}}_G + \mathcal{ANL}_G}{\mathcal{L}_G + \widetilde{\mathcal{L}}_G + \mathcal{NL}_G} \right) = \left(\frac{1+1+1}{1+2+1} \right) = \left(\frac{3}{4} \right) = 0.75$

- $h_{\widetilde{GC}}(Goal\ 2) = \left(\frac{\mathcal{AL}_G + \widetilde{\mathcal{AL}}_G + \mathcal{ANL}_G}{\mathcal{L}_G + \widetilde{\mathcal{L}}_G + \mathcal{NL}_G} \right) = \left(\frac{1+0+1}{1+1+1} \right) = \left(\frac{2}{3} \right) = 0.66$

Thus, according to the scores computed by $h_{\widetilde{GC}}$, the goal with the highest score is Goal 1.

We now present the computational complexity of the *Enhanced Goal Completion Heuristic $h_{\widetilde{GC}}$*. Besides extracting *definite* and *possible* landmarks for every candidate goal (EL), this heuristic iterates over the set of candidate goals $\mathcal{G}$, and the observations sequence Obs. During the iteration over the observation sequence Obs, we extract *overlooked* landmarks from the observations Obs (EOL). The heuristic also iterates over the extracted landmarks ($\mathcal{L}_G + \widetilde{\mathcal{L}}_G + \mathcal{NL}_G$) to compute the achieved landmarks. The heuristic computation of $h_{\widetilde{GC}}$ (HC) is linear on the number of landmarks. Thus, the complexity of this heuristic approach is: $O(EL + |\mathcal{G}| \cdot |Obs| \cdot |EOL| \cdot |\mathcal{L}_G + \widetilde{\mathcal{L}}_G + \mathcal{NL}_G| + HC)$.

3.3.2 Enhanced Uniqueness Heuristic for Incomplete Domain Models

Most goal recognition problems contain multiple candidate goals that share common fact landmarks, generating ambiguity to recognize correctly the intended goal. Evidently, landmarks that are common to multiple candidate goals are less useful for recognizing a goal than landmarks that exist for only a single goal. Thus, computing how unique (and thus informative) each landmark is can help disambiguate similar goals for a set of candidate goals. We now develop a second goal recognition heuristic based on this intuition. To develop this heuristic, we leverage the concept of *landmark uniqueness* (introduced in APPENDIX A), which is the inverse frequency of a landmark among the landmarks found in a set of candidate goals. For example, consider a landmark L that occurs only for a single goal within a set of candidate goals; the uniqueness value for such a landmark is intuitively the maximum value of 1. Equation 3.2 formalizes this intuition, describing how the *landmark uniqueness value* is computed for a landmark L and a set of landmarks for all candidate goals $K_{\mathcal{G}}$.

$$L_{Uniq}(L, K_{\mathcal{G}}) = \left(\frac{1}{\sum_{\mathcal{L} \in K_{\mathcal{G}}} |\{L|L \in \mathcal{L}\}|} \right) \tag{3.2}$$

We use the concept of *landmark uniqueness value* to estimate which candidate goal is the intended one by summing the uniqueness values of the landmarks achieved in the observations. Unlike our previous heuristic (Equation 3.1), which estimates progress towards goal completion by analyzing just the set of achieved landmarks, the landmark–based uniqueness heuristic estimates the goal completion of a candidate goal G by calculating the ratio between

the sum of the uniqueness value of the achieved landmarks of G and the sum of the uniqueness value of all landmarks of a goal G. To build the uniqueness heuristic over *incomplete domain models*, we enhance the original uniqueness heuristics (h_{uniq}) presented in APPENDIX A by using the concepts of *definite*, *possible*, and *overlooked* landmarks. We store the set of *definite* and *possible* landmarks of a goal G separately into $\mathcal{L}_G$ and $\widetilde{\mathcal{L}}_G$, and the set of *overlooked* landmarks into $\mathcal{NL}_G$. Thus, the uniqueness heuristic effectively weighs the completion value of a goal by the informational value of a landmark so that unique landmarks have the highest weight. To estimate goal completion using the *landmark uniqueness value*, we calculate the uniqueness value for every extracted (*definite*, *possible*, and *overlooked*) landmark in the set of landmarks of the candidate goals using Equation 3.2. Since we use three types of landmarks and they are stored in three different sets, we compute the landmark uniqueness value separately for them, storing the landmark uniqueness value of *definite* landmarks $\mathcal{L}_G$ into $\Upsilon_{\mathcal{L}}$, the landmark uniqueness value of *possible* landmarks $\widetilde{\mathcal{L}}_G$ into $\Upsilon_{\widetilde{\mathcal{L}}}$, and the landmark uniqueness value of *overlooked* landmarks $\mathcal{NL}_G$ into $\Upsilon_{\mathcal{NL}_G}$. Our *Enhanced Uniqueness Heuristic* is denoted as $h_{\widetilde{UNIQ}}$ and formally defined in Equation 3.3.

$$h_{\widetilde{UNIQ}}(G) = \left(\frac{\sum\limits_{\mathcal{A}_L \in \mathcal{AL}_G} \Upsilon_{\mathcal{L}}(\mathcal{A}_L) + \sum\limits_{\widetilde{\mathcal{A}_L} \in \widetilde{\mathcal{AL}}_G} \Upsilon_{\widetilde{\mathcal{L}}}(\widetilde{\mathcal{A}_L}) + \sum\limits_{\mathcal{ANL} \in \mathcal{ANL}_G} \Upsilon_{\mathcal{NL}_G}(\mathcal{ANL})}{\sum\limits_{L \in \mathcal{L}_G} \Upsilon_{\mathcal{L}}(L) + \sum\limits_{\tilde{L} \in \widetilde{\mathcal{L}}_G} \Upsilon_{\widetilde{\mathcal{L}}}(\tilde{L}) + \sum\limits_{\mathcal{NL} \in \mathcal{NL}_G} \Upsilon_{\mathcal{NL}_G}(\mathcal{NL})} \right) \tag{3.3}$$

To exemplify how our enhanced uniqueness heuristic $h_{\widetilde{UNIQ}}$ works for recognizing goals, we use and follow the same example we used before for our previous heuristics, as shown in Example 3.4.

Example 3.4. *Considering the goal recognition problem and the sets of landmarks presented in Figure 3.3, we have the following landmark uniqueness values for the landmarks:*

- *Based on the set of landmarks for all goals $K_{\mathcal{G}} = \{a, b, c, d\}$, the landmark uniqueness values are:*

  - *$L_{Uniq}(a, K_{\mathcal{G}}) = 1/2 = 0.5$*
  - *$L_{Uniq}(b, K_{\mathcal{G}}) = 1/2 = 0.5$*
  - *$L_{Uniq}(c, K_{\mathcal{G}}) = 1/2 = 0.5$*
  - *$L_{Uniq}(d, K_{\mathcal{G}}) = 1/1 = 1$*

- *After computing the landmark uniqueness value for the set of landmarks, we now can compute the scores for Goal 1 and Goal 2 using $h_{\widetilde{UNIQ}}$, as follows:*

  - $h_{\widetilde{UNIQ}}(Goal\ 1) = \left(\frac{0.5+1+0.5}{0.5+1.5+0.5}\right) = \left(\frac{2}{2.5}\right) = 0.80$
  - $h_{\widetilde{UNIQ}}(Goal\ 2) = \left(\frac{0.5+0+0.5}{0.5+0.5+0.5}\right) = \left(\frac{1}{1.5}\right) = 0.66$

Thus, according to the scores computed by $h_{\widetilde{UNIQ}}$, the goal with the highest score is Goal 1.

We now formalize the computational complexity of the *Enhanced Uniqueness Heuristic* $h_{\widetilde{UNIQ}}$. Similar to the previous enhanced heuristic approach, this heuristic first extracts *definite* and *possible* landmarks for every candidate goal (EL), and then iterates over the set of candidate goals $\mathcal{G}$, and the observations sequence Obs. *Overlooked* landmarks are extracted (EOL) during the iteration over the observation sequence Obs. This heuristic also iterates over the extracted landmarks ($\mathcal{L}_G + \widetilde{\mathcal{L}}_G + \mathcal{NL}_G$) to compute the achieved landmarks. Different than the other enhanced heuristic, in this heuristic, we weight each landmark by how common this landmark is across all goal hypotheses. We call this weight the *uniqueness value* ($CLUniq$), and its computation is linear on the number of fact landmarks. The heuristic computation of $h_{\widetilde{UNIQ}}$ (HC) is also linear on the number of landmarks. Thus, the complexity of this heuristic approach is: $O(EL + |\mathcal{G}| \cdot |Obs| \cdot |EOL| \cdot |\mathcal{L}_G + \widetilde{\mathcal{L}}_G + \mathcal{NL}_G| + CLUniq + HC)$.

3.4   Experiments and Evaluation

In this section, we describe the experiments carried out to evaluate our goal recognition heuristics over incomplete domain models, describing how we have built and modified datasets from literature (Sections 3.4.1 and 3.4.2), as well as describing the metrics we used for evaluation (Section 3.4.3). In Section 3.4.4, we present the average number of possible complete domains models over the datasets we built with incomplete domains models, showing the complexity of recognizing goals over incomplete domain models. We compare the recognition performance of our enhanced heuristics (Sections 3.4.5 and 3.4.6) against the two original landmark–based heuristic approaches APPENDIX A, which we use as baselines. We also perform an ablation study that evaluates the effect and impact of the various types of landmarks on the recognition performance of our heuristic approaches.

3.4.1 Domains

We empirically evaluated our goal recognition heuristics by using fifth-ten domains from the planning literature[4]. Six of these planning domains have been also used in the evaluation of other goal and plan recognition approaches [77, 78, 19, 87, 70]. Specifically, we evaluate our heuristics over a variety of different types of planning domains, i.e., navigation domains, logistic-based domains, plan-library adapted domains, among others. We summarize these domains as follows.

- BLOCKS-WORLD (BLOCKS) is a domain that consists of a set of blocks, a table, and a robot hand. Blocks can be stacked on top of other blocks or on the table. A block that has nothing on it is clear. The robot hand can hold one block or be empty. The goal is to find a sequence of actions that achieves a final configuration of blocks;

- CAMPUS is a domain that consists of finding what activity is being performed by a student from his observations on a campus environment;

- DEPOTS is a domain that combines transportation and stacking. For transportation, packages can be moved between depots by loading them on trucks. For stacking, hoists can stack packages on palettes or other packages. The goal is to move and stack packages by using trucks and hoists between depots;

- DRIVER-LOG (DRIVER) is a domain that consists of drivers that can walk between locations and trucks that can drive between locations. Walking from locations requires traversal of different paths. Trucks can be loaded with or unloaded of packages. Goals in this domain consists of transporting packages between locations;

- DOCK-WORKER-ROBOTS (DWR) is a domain that involves a number of cranes, locations, robots, containers, and piles, in which goals involve transporting containers to a final destination according to a desired order;

- IPC-GRID domain is a domain consists of an agent that moves in a grid from connected cells to others by transporting keys in order to open locked locations;

- FERRY is a domain that consists of set of cars that must be moved to desired locations using a ferry that can carry only one car at a time;

---

[4]`http://ipc.icaps-conference.org`

- INTRUSION-DETECTION (INTRUSION) represents a domain where a hacker tries to access, vandalize, steal information, or perform a combination of these attacks on a set of servers;

- KITCHEN is a domain that consists of home-activities, in which the goals can be preparing dinner, breakfast, among others;

- LOGISTICS is a domain which models cities, and each city contains locations. These locations are airports. For transporting packages between locations, there are trucks and airplanes. Trucks can drive between cities. Airplanes can fly between airports. The goal is to get and transport packages from locations to other locations;

- MICONIC is a domain that involves transporting a number of passengers using an elevator to reach destination floors;

- ROVERS is a domain that consists of a set of rovers that navigate on a planet surface in order to find samples and communicate experiments;

- SATELLITE is a domain that involves using one or more satellites to make observations, by collecting data and down-linking the data to a desired ground station;

- SOKOBAN is a domain that involves an agent whose goal is to push a set of boxes into specified goal locations in a grid with walls; and

- ZENO-TRAVEL (ZENO) is a domain where passengers can embark and disembark onto aircraft that can fly at two alternative speeds between locations.

3.4.2 Datasets and Setup

For experiments and evaluation, we used and modified openly available goal and plan recognition datasets [66][5], which contain thousands of recognition problems. These datasets contain large and non-trivial planning problems (with optimal and sub-optimal plans as observations, i.e., optimal and sub-optimal behaviors) for the fifth-ten planning domains described in the previous section, including domains and problems from datasets that were developed by Ramírez and Geffner [77, 78][6]. All planning domains in these datasets are encoded using the STRIPS fragment of PDDL [51]. Each goal and plan recognition problem in these datasets

---

[5]https://doi.org/10.5281/zenodo.825878
[6]https://sites.google.com/site/prasplanning

contains a (complete) domain definition, an initial state, a set of candidate goals, a correct hidden goal in the set of candidate goals, and an observation sequence. An observation sequence contains actions that represent an optimal plan or sub-optimal plan that achieves a correct hidden goal, and this observation sequence can be full or partial. A full observation sequence represents the whole plan that achieves the hidden goal, i.e., 100% of the actions having been observed. A partial observation sequence represents a plan for the hidden goal, varying in 10%, 30%, 50%, or 70% of its actions having been observed. To evaluate our goal recognition approaches over incomplete domain models, we modify the (complete) domain models of these datasets by following the formalism of incomplete STRIPS, adding annotated possible preconditions and effects (add and delete lists). Thus, the only modification to the original datasets is the generation of new, incomplete, domain models for each recognition problem, varying the percentage of incompleteness (possible preconditions and effects) in these domains.

To build goal recognition datasets with incomplete domain models, we vary the percentage of incompleteness of a domain from 20 to 80 percent (i.e., 20%, 40%, 60%, and 80%). For example, consider that a complete domain has, for all its actions, a total of 10 preconditions, 10 add effects, and 10 delete effects. A derived model with 20% of incompleteness needs to have 2 possible preconditions (8 known preconditions), 2 possible add effects (8 known add effects), and 2 possible delete effects (8 known delete effects), and so on for other percentages of incompleteness. Like [61, 60], we used the following conditions to generate incomplete domain models with possible preconditions, possible add effects, and possible delete effects:

1. We randomly move a percentage of known preconditions and effects into possible lists of preconditions and effects;

2. We randomly add possible preconditions from delete effects that are not preconditions of a corresponding operator; and

3. We randomly add into possible lists (of preconditions, add effects, or delete effects) predicates whose parameters fit into the operator signatures and are not precondition or effects of the operator.

By following all these three conditions, we generated three different incomplete STRIPS domain models from a complete STRIPS domain model, since the lists of preconditions and effects are generated randomly. Thus, each percentage of domain incompleteness has three domain models with different possible lists of preconditions and effects.

We ran all sets of experiments using a single core of a 12 core Intel(R) Xeon(R) CPU E5-2620 v3 @ 2.40GHz with 16GB of RAM in a Linux environment using Java. The JavaVM ran experiments with a 2GB memory limit and a 2-minute time limit.

3.4.3 Evaluation Metrics

We evaluate our heuristic approaches using the standard metrics of *Precision* (ratio of correct positive predictions among all predictions) and *Recall* (ratio between true positive results and total true positive and false negative results). In order to present a unified metric, we report the *F1-score* (harmonic mean) of *Precision* and *Recall*. To perform our ablation study, we use the *Correlation* (C) between the averages of *F1-score* (F_1) and the absolute number of the various types of landmarks (*definite* D, *possible* P, and *overlooked* O landmarks) over all domains and problems, and *Spread in* $\mathcal{G}$ as S, representing the average number of returned (recognized) goals. We decided to use the *Correlation* in order to show the impact of each type of landmark in the *F1-score* over the evaluated goal recognition problems. More specifically, we aim to show the association (or relationship) between the *F1-score* and the number of extracted landmarks. *Correlation* is a real value in $[-1, 1]$ such that -1 represents an *anti-correlation* between the landmarks and the *F1-score*, 0 represents no correlation between the landmarks and the *F1-score*, whereas a value of 1 represents that more landmarks correlate to a higher *F1-score*.

Besides these metrics, we use a graphical plot to evaluate accuracy performance of our heuristic approaches over incomplete domain models. To do so, we adapt the *Receiver Operating Characteristic* (ROC) curve metric to highlight the trade-off between true positive and false positive results. A ROC curve is often used to compare not only true positive predictions, but also to compare the false positive predictions of the experimented approaches. Here, each prediction result of our goal recognition approaches represents one point in the space, and thus, instead of a curve, our graphs show the spread of our results over ROC space. In the ROC space, the diagonal line represents a random guess to recognize a goal from observations. This diagonal line divides the ROC space in such a way that points above the diagonal represent good classification results (better than random guess), whereas points below the line represent poor results (worse than random guess). The best possible (perfect) prediction for recognizing goals are points in the upper left corner (i.e., coordinate x = 0 and y = 100) in ROC space.

3.4.4 Experimental Results: The Average Number of Possible Complete Domains

Our heuristic approaches recognize goals at very low recognition time for most incomplete planning domains and problems, taking at most 2.7 seconds, including the process of extracting landmarks, among all goal recognition problems, apart from IPC-GRID and SOKOBAN, which took substantial recognition time (for more detail, please see APPENDIX B). More specifically, only 1092 (20% of domain incompleteness) out of 4368 problems for IPC-GRID and SOKOBAN do not exceed the time limit of 2 minutes (for both our approaches and the baselines). SOKOBAN exceeds the time limit of 2 minutes for most goal recognition problems because this dataset contains large problems with a huge number of objects, leading to an even larger number of instantiated predicates and actions. For example, as domain incompleteness increases (i.e., the ratio of possible and definite preconditions and effects), the number of possible actions (moving between cells and pushing boxes) increases substantially in a grid with 9x9 cells and 5 boxes as there are very few known preconditions for several possible preconditions. As a basis of comparison, state-of-the-art planners [61, 60] for incomplete domain models take substantially more time than 2-minute timeout to generate a single plan for domains that our heuristic approaches recognize goals in less than 2 seconds. For example, CPISA [60] takes $\approx$ 300 seconds to find a plan with 25 steps in domains (e.g., SATELLITE) with 2 possible preconditions and 3 possible add effects, whereas our dataset contains much more complex incomplete domains and problems. The average number of possible complete domain models $|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle|$ is huge for several domains, showing that the task of goal recognition over incomplete domain models can be quite difficult and complex if we take into account the number of possible complete domain models. For instance, the average number of possible complete domains in this dataset varies between 9.18 (SOKOBAN with 20% of domain incompleteness) and 7.84^{15} (ROVERS with 80% of domain incompleteness). The average number of possible complete domain models $|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle|$ is huge for several domains (CAMPUS, DWR, KITCHEN, and ROVERS), showing that the task of goal recognition in incomplete domains models is quite difficult and complex. Table 3.1 shows the average number of possible complete domain models for all domains we use in our experiments.

Incompleteness of $\widetilde{\mathcal{D}}$ (%)	20%	40%	60%	80%
#	$|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle|$	$|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle|$	$|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle|$	$|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle|$
BLOCKS	42.22	1782.89	75281.09	3178688.03
CAMPUS	7131.55	50859008.46	3.63E+11	2.59E+15
DEPOTS	168.89	28526.20	4817990.10	813744135.40
DRIVER	48.50	2352.53	114104.80	5534417.30
DWR	512.00	262144.00	134217728.00	6.88E+10
FERRY	8.00	64.00	512.00	4096.00
INTRUSION	16.00	256.00	4096.00	65536.00
IPC-GRID	10.55	111.43	1176.26	12416.75
KITCHEN	2767208.65	7.66E+12	2.11E+19	5.86E+25
LOGISTICS	27.85	776.04	21618.81	602248.76
MICONIC	9.18	84.44	776.05	7131.55
ROVERS	9410.14	88550676.93	8.34E+11	7.84E+15
SATELLITE	27.85	776.04	21618.81	602248.76
SOKOBAN	9.18	84.44	776.04	7131.55
ZENO	48.50	2352.53	114104.80	5534417.30

Table 3.1: The average number of possible complete domain models $|\langle\langle\widetilde{\mathcal{D}}\rangle\rangle|$ for all domains we use in our experiments.

3.4.5 Experimental Results: An Ablation Study of The Impact of New Notions of Landmarks on The Recognition Performance

Since the key contribution of our heuristic approaches to goal recognition over incomplete domains are based on the new types of landmarks (*definite*, *possible*, and *overlooked*), as opposed to the traditional landmarks from *Classical Planning*, in this section, we want to objectively measure the effect of these new types of landmark on the recognition performance of our heuristics. Thus, we now present an ablation study that consists of measuring the performance of our heuristic approaches using some possible combinations of landmark types.

Table 3.2 summarizes the results of our ablation study by aggregating the average results over the datasets we generated for incomplete domains considering all levels of domain incompleteness (20%, 40%, 60%, and 80%) and observability (10%, 30%, 50%, 70%, and 100%). In APPENDIX B, we show in detail the results for all fifth-ten domains by varying the level of domain incompleteness and observability. We denote the original landmark–based heuristic approaches (APPENDIX A) as Baseline (h_{gc}) and Baseline (h_{uniq}). We run the experiments for the baselines ignoring the incomplete part of the domain model, i.e., *all possible preconditions and effects*. This allows us to evaluate the effectiveness of the new types of landmarks over incomplete domains in our enhanced heuristics. We denote our enhanced heuristics as $\widetilde{GC}$ and $\widetilde{UNIQ}$, and denote the combination over the various types of landmarks using $D + P + O$ as

the combination of *definite, possible*, and *overlooked* landmarks, and other four combinations of these landmark types ($D + O$, $P + O$, P, and O). For this ablation study, we report the results by evaluating our enhanced heuristic approaches and the baselines using the *F1-score* metric (F_1), the average number of recognized goals *Spread in* $\mathcal{G}$ (S), the *Correlation* between the averages of landmarks and *F1-scores* (*Correlation* of *definite* landmarks CD, *Correlation* of *possible* landmarks CP, and *Correlation* of *overlooked* landmarks CO), as well as the average number of extracted landmarks for all types (D, P, and O).

We compute the *Correlation* between the averages of landmarks and *F1-scores* over all domains and degrees of incompleteness, columns CD, CP and CO in Table 3.2, and plot these correlations as a function of the level of incompleteness in Figures 3.4a–3.4f as opposed to how traditional landmarks affect performance on the baseline approaches (*Correlation* varying between 0.32 and 0.67 for a lower *F1-score*, Table 3.2, lines 1 and 2). Figures 3.4a and 3.4d show how all types of landmark correlate to the performance of heuristic approaches over incomplete domain models (represented by $\widetilde{GC}$ ($D+P+O$) and $\widetilde{UNIQ}$ ($D+P+O$)). At low levels of incompleteness (20% and 40%) we have larger *F1-scores* and larger numbers of *definite* landmarks, leaving a smaller number of *overlooked* landmarks to be inferred *on the fly*. Under these conditions, *overlooked* landmarks start off at a slight anti-correlation with performance. As the level of incompleteness of the domain description increases, the number of *definite* landmarks decreases, but their *Correlation* to performance increases. This suggests that an increase in the number of inferred landmarks leads to better performance. The number of *overlooked* landmarks remains broadly the same over time, as they are tied to the amount of information in the observations more than they are tied to the information in the domain description, and their *Correlation* to performance monotonically increases as the incompleteness increases. This indicates that *overlooked* landmarks play an increasingly important role in the recognition performance as incompleteness increases, giving more information to our enhanced heuristics. The number of *possible* landmarks also varies with domain incompleteness, initially increasing as the number of possible effects increases, to subsequently decrease as the number of possible effects leads to less bottlenecks in the state-space to yield landmarks. As the number of possible effects increases, so does their unreliability as sources of landmarks, which is reflected in their decreasing *Correlation* to performance.

As we ablate landmarks, performance drops most substantially when we remove either *definite* or *possible* landmarks from the enhanced heuristics (Figures 3.4c and 3.4f), indicating their importance to recognition accuracy. We can also see that when using *overlooked* landmarks

exclusively, in $\widetilde{GC}$ (O) and $\widetilde{UNIQ}$ (O), it provides a close approximation of the performance of the technique using all landmark types. Therefore, this is strong evidence that *overlooked* landmarks are one of the most important contributions of this thesis.

Figure 3.5 compares all evaluated approaches with respect to *F1-score* averages, varying the domain incompleteness from 20% to 80%. Thicker lines represent the recognition approaches that have higher *F1-scores*. Note that our enhanced heuristics ($\widetilde{GC}$ and $\widetilde{UNIQ}$) that combine the use of the new types of landmarks, i.e., *D+O*, *D+P+O*, and *O* are the approaches that have the higher *F1-scores* over the datasets, showing that using *overlooked* landmarks substantially improves goal recognition accuracy.

	Domain Incompleteness 20%						Domain Incompleteness 40%						Domain Incompleteness 60%						Domain Incompleteness 80%					
	$|D|$	$|P|$	$|O|$	$|S|$	$|F_1|$	$CD/CP/CO$	$|D|$	$|P|$	$|O|$	$|S|$	$|F_1|$	$CD/CP/CO$	$|D|$	$|P|$	$|O|$	$|S|$	$|F_1|$	$CD/CP/CO$	$|D|$	$|P|$	$|O|$	$|S|$	$|F_1|$	$CD/CP/CO$
Baseline (h_{gc})	11.5	0	0	1.53	0.44	0.35/0/0	4.9	0	0	2.27	0.36	0.42/0/0	3.5	0	0	3.32	0.31	0.65/0/0	2.9	0	0	4.85	0.34	0.67/0/0
Baseline (h_{uniq})	11.5	0	0	1.48	0.44	0.32/0/0	4.9	0	0	1.95	0.34	0.36/0/0	3.5	0	0	2.83	0.32	0.33/0/0	2.9	0	0	4.12	0.33	0.40/0/0
$\widetilde{GC}$ (D+P+O)	11.6	1.4	10.1	1.34	0.75	0.22/0.13/-0.51	9.6	2.1	9.6	1.46	0.74	0.56/0.23/-0.20	7.1	2.3	9.4	1.75	0.68	0.73/0.41/0.16	6.8	1.2	9.3	1.98	0.65	0.61/0.02/0.33
$\widetilde{GC}$ (D+O)	11.6	0	12.3	1.37	0.74	0.23/0/-0.42	9.6	0	11.9	1.54	0.73	0.57/0/-0.11	7.1	0	12.8	1.88	0.68	0.76/0/0.37	6.8	0	10.9	2.10	0.64	0.62/0/0.35
$\widetilde{GC}$ (P+O)	0	1.4	20.7	5.22	0.31	0/-0.13/-0.44	0	2.1	16.8	4.77	0.34	0/-0.06/-0.30	0	2.3	14.2	4.13	0.33	0/0.25/-0.22	0	1.2	13.3	4.87	0.27	0/-0.20/-0.03
$\widetilde{GC}$ (P)	0	1.4	0	2.14	0.22	0/0.29/0	0	2.1	0	2.12	0.28	0/0.07/0	0	2.3	0	2.09	0.26	0/0.56/0	0	1.2	0	1.85	0.18	0/0.06/0
$\widetilde{GC}$ (O)	0	0	23.1	1.88	0.71	0/0/-0.21	0	0	20.0	2.02	0.70	0/0/0.05	0	0	17.4	2.30	0.65	0/0/0.42	0	0	15.1	2.46	0.63	0/0/0.34
$\widetilde{UNIQ}$ (D+P+O)	11.6	1.4	10.1	1.32	0.71	0.21/0.23/-0.37	9.6	2.1	9.6	1.41	0.70	0.28/0.21/-0.10	7.1	2.3	9.4	1.65	0.64	0.82/0.59/0.31	6.8	1.2	9.3	1.90	0.62	0.61/0.034/0.41
$\widetilde{UNIQ}$ (D+O)	11.6	0	12.3	1.34	0.68	0.35/0/-0.26	9.6	0	11.9	1.52	0.65	0.38/0/0.11	7.1	0	12.8	1.81	0.61	0.66/0/0.41	6.8	0	10.9	2.10	0.62	0.63/0/0.34
$\widetilde{UNIQ}$ (P+O)	0	1.4	20.7	5.23	0.32	0/-0.14/-0.52	0	2.1	16.8	4.96	0.35	0/-0.19/-0.53	0	2.3	14.2	4.80	0.33	0/0.10/-0.33	0	1.2	13.3	5.31	0.29	0/-0.57/-0.45
$\widetilde{UNIQ}$ (P)	0	1.4	0	2.05	0.28	0/-0.27/0	0	2.1	0	2.00	0.27	0/-0.42/0	0	2.3	0	1.99	0.27	0/-0.19/0	0	1.2	0	1.81	0.27	0/-0.55/0
$\widetilde{UNIQ}$ (O)	0	0	23.1	1.75	0.69	0/0/-0.11	0	0	20.0	1.91	0.77	0/0/0.03	0	0	17.4	2.16	0.63	0/0/0.39	0	0	15.1	2.35	0.61	0/0/0.33

Table 3.2: Experimental results for our ablation study, comparing the baseline approaches h_{gc} and h_{uniq} (APPENDIX A) against our enhanced heuristics using various combinations of landmark types. This table aggregates the average results for all metrics over all datasets for all levels of observability.

In APPENDIX B we report detailed results for all evaluated approaches and domains, varying not only domain incompleteness but also the percentage of observability of the observation sequence, showing the averages for all types of landmarks, *F1-score*, and *Correlation*. Namely, each inner table in Tables APPENDIX B.3 and APPENDIX B.4 summarize the results for each percentage of observability (10%, 30%, 50%, 70%, and 100%) over the evaluated datasets. Thus, it is possible to see that the combination of all notions of landmarks $(D+P+O)$ when applied for both enhanced heuristics outperforms the other combinations, including the baseline approaches (APPENDIX A), in all variations of domain incompleteness and observability.
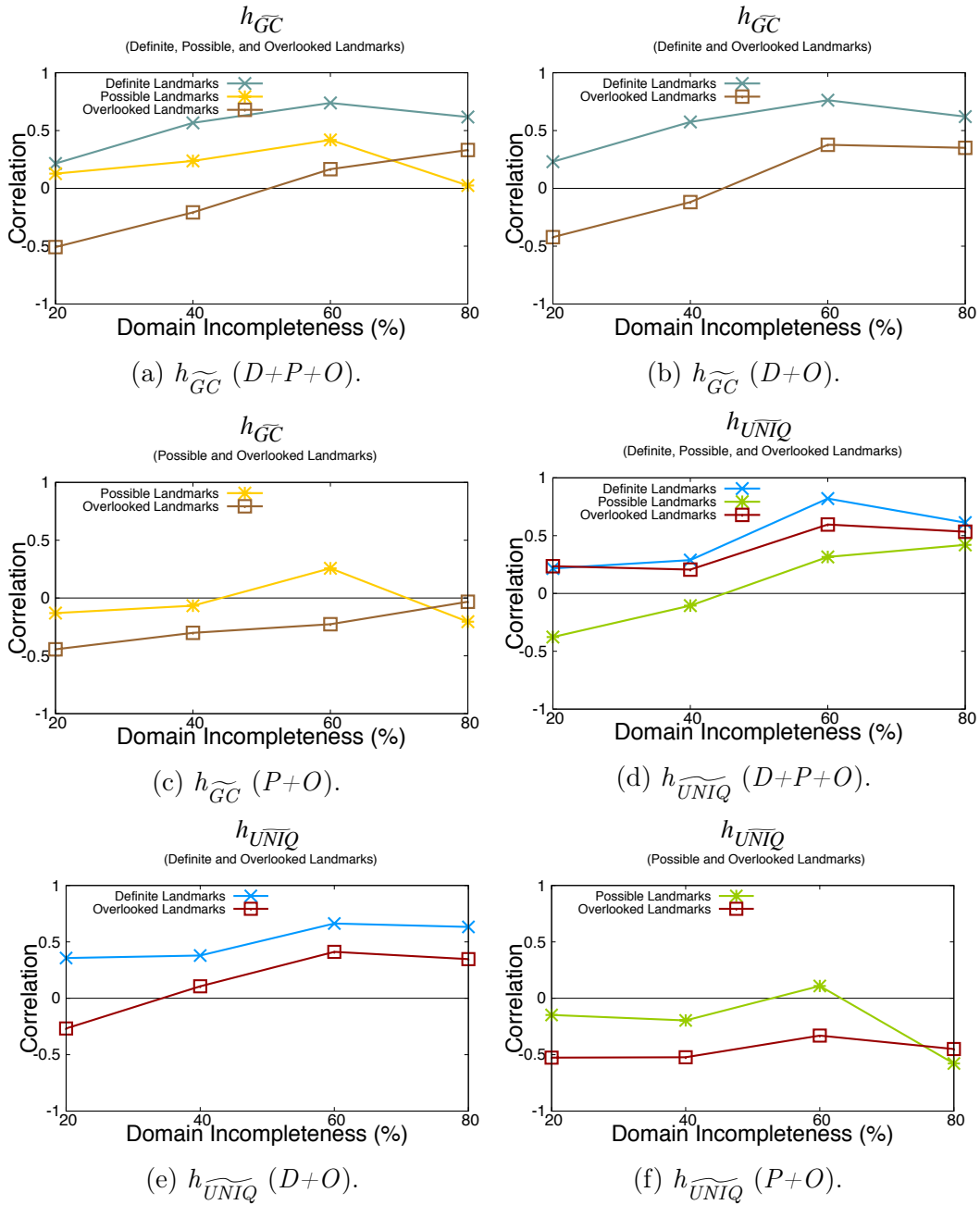
Figure 3.4: *Correlation* of landmarks to performance (*F1-score*). .

F1-Score Comparison
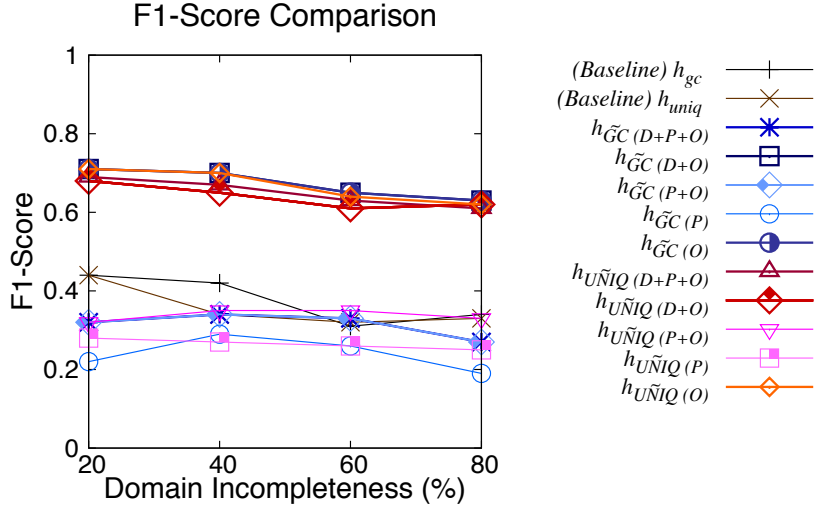
Figure 3.5: *F1-score* average of all evaluated approaches over the datasets varying the domain incompleteness.

3.4.6   Experimental Results: ROC Space Analysis

We now present our second set of experiments, comparing the results of our enhanced heuristic approaches against the baselines(h_{gc} and h_{uniq}) (APPENDIX A) using ROC space, which shows the trade-off between true positive and false positive results. The use of ROC space allows us to see graphically what approach is more accurate for recognizing goals over the datasets we used. Therefore, the approach that has more points in the upper left corner (i.e., true positive rate equals to 100%, and false positive rate equals to 0%) is the most accurate approach over the used datasets. Figure 3.6 shows ROC space graphs corresponding to recognition performance over the four percentages of domain incompleteness we used in our experiments. We aggregate multiple recognition problems for all domains and plot these results in ROC space varying the percentage of domain incompleteness.

We report the results of our enhanced heuristics ($h_{\widetilde{GC}}$ and $h_{\widetilde{UNIQ}}$) in Figure 3.6 by using the combination of landmarks that has the best results when applied to our heuristics (i.e., using *definite*, *possible*, and *overlooked* landmarks, as shown in the previous section), against the baselines h_{gc} and h_{uniq} (APPENDIX A), that uses just the landmarks extracted by a traditional landmark extraction algorithm, i.e., ignoring the incomplete part of the domain model (possible preconditions and effects). Although the true positive rate is high for most recognition problems at most percentages of domain incompleteness, as the percentage of domain incompleteness increases, the false positive rate also increases, leading to several problems being recognized with a performance close to the random guess line. This happens because the number of extracted
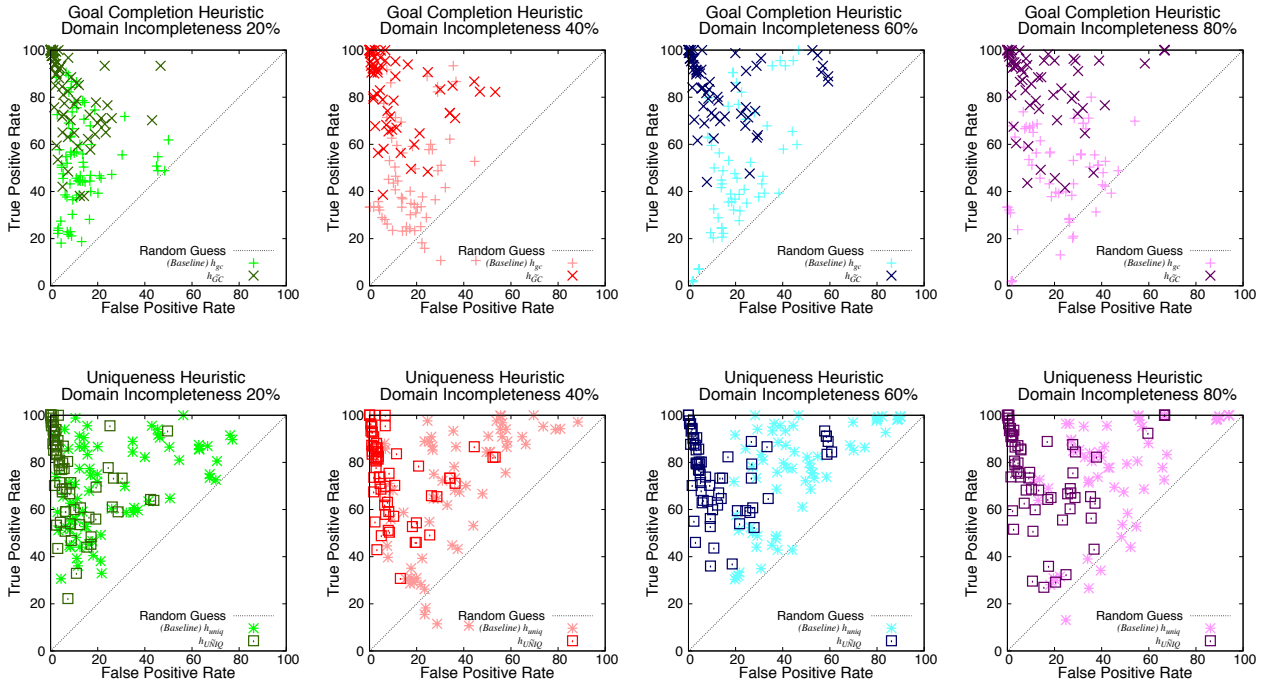
Figure 3.6: ROC space analysis for all four percentage of domain incompleteness, comparing our enhanced heuristic approaches against the baselines h_{gc} and h_{uniq} (APPENDIX A).

landmarks decreases significantly as the number of known preconditions and effects diminishes, and consequently, all candidate goals have few (if any) landmarks. For example, in several cases in which domain incompleteness is 60% and 80%, the set of landmarks is quite similar, leading our enhanced heuristics to return more than one candidate goal as the correct one. Thus, there are more returned goals during the recognition process as incompleteness increases. These results show that our enhanced heuristics perform better and are more accurate than the baselines. It is possible to see that both our enhanced heuristics aggregate most points in the left corner, while the points for the baseline approaches are closer to (and sometimes below) the random guess line.

3.5   Chapter Remarks

In this chapter, we developed novel goal recognition heuristic approaches that cope with incomplete domain models that represent possible preconditions and effects besides traditional complete models where such information is assumed to be known. We developed novel recognition heuristics by exploiting the new notions of landmarks over incomplete domain models. Our new notions of landmarks include that of *possible* landmarks for incomplete domains as well as *overlooked* landmarks that allow us to compensate fast but non-exhaustive landmark extraction algorithms, the latter of which can also be employed to improve existing goal and plan recognition approaches [65, 70]. Experiments over thousands of goal recognition problems in fifth-ten planning domain models show two key results of our enhanced heuristics. First, these enhanced heuristics are fast and accurate when dealing with incomplete domains at all variations of observability and domain incompleteness. The use of novel heuristics frees us from using full-fledged incomplete-domain planners as part of the recognition process. Approaches that use planners for goal recognition are already very expensive for complete domains and are even more so in incomplete domains, since they often generate plans taking into consideration many of the possible models, and even then they often fail to generate robust plans for these domains. Second, our ablation study shows that our new notions of landmarks have a substantial impact on the accuracy of our heuristics over simply ignoring the uncertain information from the domain model, as we use in the baseline approaches. Importantly, the ablation study shows that overlooked landmarks contribute substantially to the accuracy of our approach. As future work, we envision such techniques to be instrumental in using learned planning models [5] for goal recognition [3].

In summary, we have presented in this chapter the following contribution to the goal and plan recognition community.

1. We extended the goal and plan recognition problem introduced by Ramírez and Geffner in [77, 78], and defined a new problem for recognizing goals over incomplete domains;

2. We introduced new notions of landmarks for incomplete domains models, i.e., *definite*, *possible*, and *overlooked* landmarks;

3. We developed a novel algorithm to extract these new notions of landmarks over incomplete domains; and

4. We enhanced landmark–based heuristics from the literature [70] to cope with incomplete domains and use our notions of landmarks, and showed that these notions of landmarks have a significant impact on the recognition performance over incomplete domain models.

4. GOAL RECOGNITION OVER NOMINAL MODELS

Existing model–based approaches to goal and plan recognition rely on expert knowledge to produce symbolic descriptions of the dynamic constraints domain objects are subject to, and these are assumed to yield correct predictions. In this chapter, we develop goal recognition approaches (Sections 4.4 and 4.5) that drop this assumption, and consider the use of *nominal models* that we can be *learned* from observations on transitions from systems with unknown dynamics. Leveraging existing work on the acquisition of domain models via *Deep Learning* for *Hybrid Planning* [85] we adapt and evaluate existing goal recognition approaches [78, 92, 39] to analyze how prediction error, inherent to system dynamics identification and model learning techniques, have an impact over recognition error rates. We evaluate the proposed recognition approaches over *nominal models* empirically in Section 4.6, using three benchmark domains based on the *constrained* Linear–Quadratic Regulator (LQR) problem [9], with increasing dimensions of state and action spaces, and two variations of a non–linear navigation domain proposed by Say et al. in [85].

4.1 Problem Formulation

Assuming the availability of complete and correct models is considered a strong assumption by the literature of *Control* and *Robotics* [56], especially when dealing with real-world and practical applications [37, 32, 45, 13], where *actual model* parameters are usually unknown, and sometimes these parameters may change over time due to wear and tear of the physical components of a robot or autonomous vehicles. The state transition of the underlying system dynamics can be obtained from observations on the behavior of other agents [13], random excitation, or the simulation of plans and control trajectories derived from *actual models* (Definition 11). In this thesis, we adopt this stance to define the task of goal recognition over *nominal models*, which models that are estimated (or learned) from past observed state transitions (Definition 12).

We formally define, in Definition 19, the task of goal recognition over Finite-Horizon Optimal Control (FHOC) problems (Section 2.5.1) and *nominal models* (Definition 12) by following the formalism of Ramírez and Geffner [77, 78], as follows.

Definition 19 (**Goal Recognition Problem over a Nominal Model**). *A goal recognition problem over a nominal model is given by:*

- *An estimated transition function $\hat{f}(x_k, u_k, w_k)$, such that, $\hat{f}(x_k, u_k, w_k) = \hat{x}_{k+1}$, where x_k is a state, u_k a control input, and w_k is a random variable;*

- *A cost function J;*

- *An initial state $\mathcal{I}$, i.e., an arbitrary element of the set of states S;*

- *A set of hypothetical candidate goals $\mathcal{G}$, including a correct hidden goal G^* (i.e., $G^* \in \mathcal{G}$);*

- *A sequence of observations $Obs = \langle o_1, o_2, ..., o_m \rangle$; and*

- *A horizon H (i.e., a fixed number of steps).*

For recognizing goals over *nominal* models, we define the sequence of observations Obs to be a partial trajectory of *states* $x \in S$ induced by a policy π (Equation 2.8) that minimizes the cost function J. In general, a finite but indeterminate number of intermediate states may be missing between any two observations o_i, $o_{i+1} \in Obs$.

In this thesis, we perform the task of goal recognition over *nominal models* in two forms: *online* and *offline* recognition. More specifically, we draw a distinction between *online* [8, 92] and *offline* goal recognition, in which the former is a sequence of m goal recognition problems where the observation sequence Obs is obtained incrementally, while in the later the observation sequence Obs is available immediately. We borrow the term *judgment point* from Baker et al. [8] to refer to the act of solving each of the m goal recognition problems that follow from the arrival of each new observation.

Informally, solving a goal recognition problem requires us to select a *candidate* goal $\hat{G} \in \mathcal{G}$ such that $\hat{G} = G^*$, on the basis of how well $\hat{G}$ predicts or explains the observation sequence Obs [8, 78]. Typically, this cannot be done exactly, but it is possible to produce a probability distribution [77, 78, 19, 87] over the set of hypothetical candidate goals $G \in \mathcal{G}$ and Obs, where the goals that best explain Obs are the most probable ones. We illustrate the goal recognition process over *nominal models* in Figure 4.1, following Definition 19. Note that in Figure 4.1 we refer to the approximate transition function $\hat{f}$ as "black box", since it is not directly accessible in our setting, and represented with a "black box" neural network. Next, in Section 4.2, we show how we use neural networks as *nominal models*.
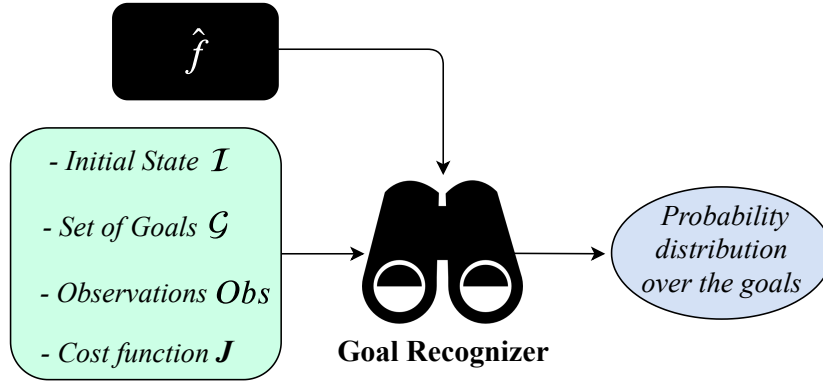
Figure 4.1: Problem overview to goal recognition over *nominal models*.

4.2   DNNs as Nominal Models

Artificial Neural Networks have shown to be very effective at learning and approximating linear and non–linear functions from data [62]. To learn and approximate the dynamics constraints in FHOC problems, and therefore, acquire *nominal models*, we leverage existing work in *Automated Planning* and *Machine Learning* that uses Deep Neural Networks (DNNs) to approximate (linear and non–linear) functions [85, 97, 84].

Learning the dynamics or the transition between states of a domain model from data can be formalized as the problem of finding the parameters θ for a function

$$\hat{f}(x_k, u_k, w_k; \theta)$$

that minimize a given *loss function* $\mathcal{L}(\mathcal{K}, \theta)$ over a dataset $\mathcal{K} = \{ (x, u, y) \mid y = f(x, u, w), y \in S \}$, where x is the state, u is the control input, w represents a random variable, and y is the resulting state after applying $f(x, u, w)$. In this thesis, we use the procedure and neural architecture reported by Say et al. [85] to acquire $\hat{f}$, namely, a DNN using Rectified Linear Units (ReLUs) [59] as the activation function, given by $h(x) = max(x, 0)$. A DNN is densely connected, and consists of L layers, $\theta = (\mathbf{W}, \mathbf{b})$, where $\mathbf{W} \in \mathbb{R}^{d \times d \times L}$ and $\mathbf{b} \in \mathbb{R}^{1 \times L}$. We use the loss function proposed by Say et al. [85], as follows:

$$\sum_{j}^{|\mathcal{D}|} ||\hat{y}_j - y_j|| + \lambda \sum_{l}^{L} ||\mathbf{W}_l||^2$$

where $\hat{y}_j = \hat{f}(x_j, u_j; \theta)$ and λ is a Tikhonov L^2-regularization hyper-parameter [30]. In the context of optimization for Machine Learning, using this regularization technique induces the

optimization algorithm to overestimate the variance of the dataset, so weights associated with unimportant directions of the gradient of $\mathcal{L}$ decay away during training. As noted by Good-fellow et al. [30], ReLU networks represent very succinctly a number of linear approximation surfaces that is exponential in the number of layers L [57]. This strongly suggests that ReLU networks can displace Gaussian process estimation [79] as a good initial choice to approximate complex non–linear stationary random processes, such as those in Equation 2.2, with the further advantage that, as demonstrated in [98, 85], DNNs can be directly used in Equation 2.6, so existing optimization algorithms can be used off–the–shelf. Figure 4.2 illustrates the neural architecture reported by Say et al. [85], in which we use to represent *nominal models*.
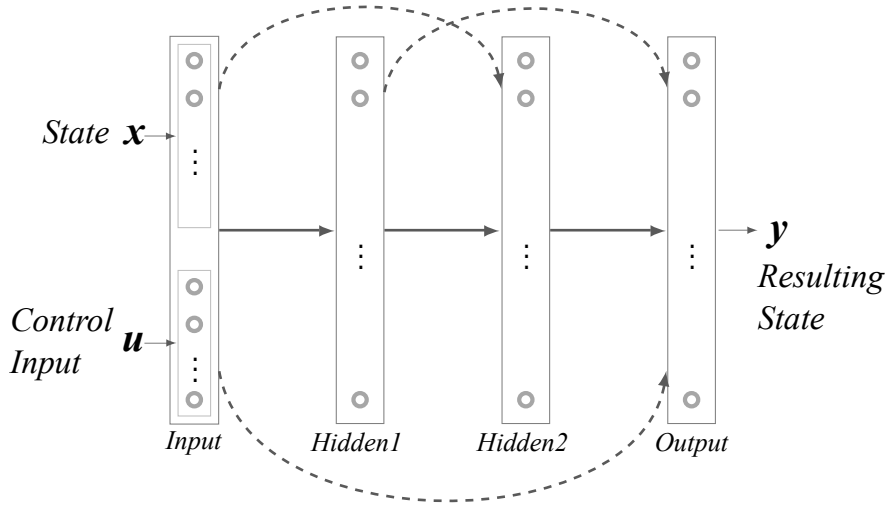
Figure 4.2: An example of a neural network with 4 layers (input layer, 2 hidden layers, and output layer), adapted from [85]. Arrows represent full connections between the input and output nodes. Every hidden layer output unit is passed through a ReLU unit. Dashed arrows are optional dense connections.

4.3   Probabilistic Goal Recognition

We follow Ramírez and Geffner [78] (R&G10) and adopt the modern probabilistic interpretation of Dennet's *principle of rationality* [18], the so–called *Bayesian Theory of Mind*, as introduced by a series of ground-breaking cognitive science studies by Baker et al. [8][88, Chapter 7]. R&G10 set the probability distribution over the set of hypothetical goals $\mathcal{G}$ and the observation sequence *Obs* introduced above to be the Bayesian posterior conditional probability

$$P(G \mid Obs) = \alpha P(Obs \mid G)P(G) \tag{4.1}$$

where $P(G)$ is a *prior* probability assigned to goal G, α is a normalization factor inversely proportional to the probability of *Obs*, and $P(Obs \mid G)$ is

$$P(Obs \mid G) = \sum_{\pi} P(Obs \mid \pi)P(\pi \mid G) \tag{4.2}$$

$P(Obs \mid \pi)$ is the probability of observing *Obs* by executing a policy (or a plan) π and $P(\pi \mid G)$ is the probability of an agent choosing plan π to achieve the goal G. Crucially, this later probability is defined to be a function that compares a measure of the efficiency of π with some suitably defined baseline of rationality, ideally the optimal plan or policy for G. In Sections 4.4 and 4.5, we discuss two well–known existing approaches to approximate Equation 4.2, both reasoning over *counterfactuals* [64] in different ways. These approaches frame in a probabilistic setting the so–called *but-for* test of causality [33]. That is, if a candidate goal G is to be considered the *cause* for observations *Obs* to happen, evidence of G being *necessary* for *Obs* to happen is required. The changes to existing approaches are motivated by us wanting to retain the ability to *compute* counterfactual trajectories when transition functions cannot be directly manipulated.

4.4   Goal Recognition as Nominal Mirroring

Mirroring [92] is an online goal recognition approach that works on both continuous and discrete domain models. For each of the candidate goal G in $\mathcal{G}$, Halpern's *but-for* test is implemented by comparing two plans: an *ideal* plan and the *observation-matching* plan (*O*-plan). Ideal plans are optimal plans computed for every candidate goal G in $\mathcal{G}$ from the initial state $\mathcal{I}$, which are *pre-computed* before the recognition process starts. The *O*-plan is also computed for every pair $(\mathcal{I}, G)$ and it is required to visit every state in the observation sequence *Obs*. *O*-plans are made of a *prefix*, that results from concatenating the *O*-plans computed for previous judgment points [8], and a *suffix*, a plan computed from the last observed state to each candidate goal G. The *but-for* test is implemented by making of use of Theorem 7 in [77], that amounts to considering a candidate G to be *necessary* for *Obs* to happen, if the cost of optimal

plans and those consistent with the observation Obs are the same. Vered et al. [92] show that O-plans are indeed consistent with Obs so Ramírez and Geffner's results apply. The test was later cast in a probabilistic framework by Kaminka et al. [39], with Equation 4.2 becoming

$$P(Obs \mid G) = [1 + \epsilon(\pi_{Obs,G}, \pi_G)]^{-1} \tag{4.3}$$

$\epsilon(\pi_{Obs,G}, \pi_G)$ above is the *matching error* of π_G, the ideal plan for G with regard to $\pi_{Obs,G}$, the O-plan for the observations. Under the assumption that w is a random variable $w \sim \mathcal{N}(0, \sigma)$ with values given by a Gaussian distribution with mean 0 and standard deviation σ, ϵ can be used to account for the influence of w as long as σ remains an order of magnitude smaller than the values given by $f(x, u, 0)$. Kaminka et al. [39] define the *matching error ϵ* as the sum of the squared errors between states in the trajectory of π_G, and those found along the trajectory of $\pi_{Obs,G}$. Under the second assumption that the selected ideal plans for a goal G are the most likely too, Kaminka et al. ϵ is an unbiased estimator for the likelihood of $\pi_{Obs,G}$.

Having established the suitability of Kaminka et al. [39] means to bring about the *but–for* test to FHOC problems, we now describe how we depart from their method to obtain O-plans $\pi_{Obs,G}$. In this thesis, for online goal recognition we construct $\pi_{Obs,G}$ by calling a planner once for each *new* observation o added to Obs *and* candidate goal G, rather than just once per candidate goal G as proposed by Kaminka et al. [39]. In doing so, it allows us to enforce consistency with observations Obs, since the couplings between states, inputs and perturbation in $\hat{f}(x, u, w)$ are no longer available so we can influence them with additional constraints, but are rather "hidden" in the network parameters. As the first observation o_1 is obtained, we call a planner to solve Equations 2.4–2.7 (from Chapter 2), setting the initial state x_0 to $\mathcal{I}$ and x_N to x_{o_1}, the state embedded in o_1. The resulting trajectory m_{Obs}^1 is then used to initialize $\pi_{Obs,G}^- = \langle m_{Obs}^1 \rangle$. We then invoke the planner again, this time setting $x_0 = x_{o_1}$ and some suitably defined constraints such that $x_N \in S_G$ for every candidate goal G. The resulting trajectories m_G^1 are used to define the O-plans $\pi_{Obs,G} = \pi_{Obs,G}^- \oplus m_G^1$, which are compared with the pre-computed ideal plans π_G to evaluate $P(Obs|G)$, according to Equation 4.3. As further observations o_i, $i > 1$, are received, we obtain trajectories m_{Obs}^i as above but setting $x_0 = x_{o_{i-1}}$ and x_N to x_{o_i}, which are used to update $\pi_{Obs,G}^-$ setting $\pi_{Obs,G}^- = \pi_{Obs,G}^- \oplus m_{Obs}^i$. Trajectories m_G^i are obtained by setting initial states to x_{o_i}, and concatenated to the updated $\pi_{Obs,G}^-$ to obtain the O-plan for the i-th judgment point. As an extension of the original *Mirroring* approach developed by

Vered et al. [92], and an adaptation of the probabilistic framework of Kaminka et al. [39], we call this approach as *Nominal Mirroring*, and denote it as ηMIRRORING.

Thus, given the plans $\pi_{Obs,G}$ and π_G for every goal $G \in \mathcal{G}$, we can then try to maximize $P(Obs \mid \pi)$, calculating the *matching error* $\epsilon(\pi_{Obs,G}, \pi_G)$ for these two plans. To calculate the *matching error* ϵ, we use a state-distance metric E over the plans $\pi_{Obs,G}$ and π_G. Here, we use the Euclidean distance as the state-distance metric E, like Kaminka et al. have used for goal recognition over continuous domains in [39]. As a result, the best matching error for $\epsilon(\pi_{Obs,G}, \pi_G)$ is 0 when these plans are identical (i.e., state-variables x_k for both plans with exactly the same values) according to the state-distance metric E. Therefore, if $\epsilon(\pi_{Obs,G}, \pi_G) = 0$, then $P(Obs \mid \pi)$ is equal to 1 (Equation 4.3).

Figure 4.3 illustrates graphically how we calculate the *matching error* ϵ between ideal plans and O-plans for recognizing goals over *nominal models*. For example, consider that the *matching error* ϵ between the O-plan $\pi_{Obs,G}$ and the ideal π_G in Figure 4.3 is 0.5, we can see that the plans are similar, and thus, we can use ϵ to compute $P(Obs|\pi)$ (Equation 4.3), as follows: $P(Obs \mid \pi) = [1 + 0.5]^{-1} = 0.66$.
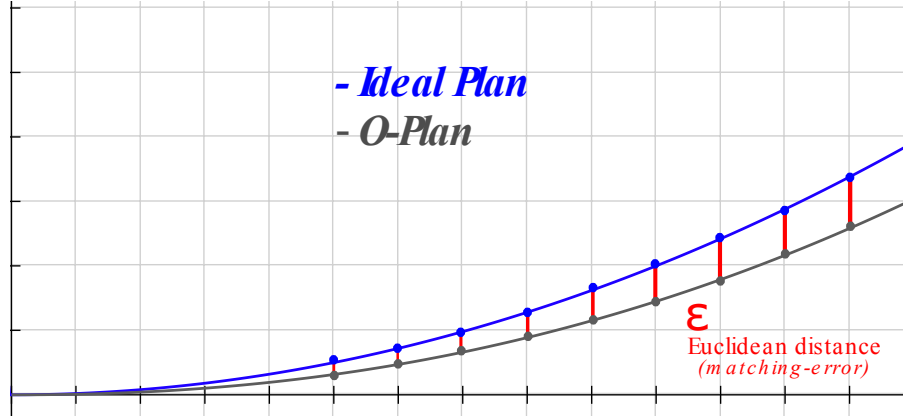
Figure 4.3: *Nominal Mirroring* approach example.

To analyze the computational complexity of ηMIRRORING, we use as a baseline Vered et al.'s [92] original *Mirroring* approach, which requires $|\mathcal{G}|$ calls to a planner per observation, and $|\mathcal{G}| + (|\mathcal{G}||Obs|)$ calls overall. In contrast, our ηMIRRORING requires $|\mathcal{G}| + 1$ calls to a planner per observation, and $|\mathcal{G}| + |Obs| + (|\mathcal{G}||Obs|)$ calls overall. This is a slight overhead which, on the basis of the results in Section 4.6, seems to be amortized enough by the accuracy and robustness of our new method.

4.5   Goal Recognition Based on Cost Differences

We now develop a novel goal recognition approach over *nominal models* based on cost differences, inspired by the probabilistic framework of R&G10. In [78], R&G10 implement Halpern's *but-for* test [33] by determining whether plans exist that, while achieving G, either guarantee that Obs happens, or prevent it from happening, the later being the *counterfactual* plan [64]. When no such plans exist, we obtain proof of G not being *sufficient* cause for Obs. Typically though, goals remain feasible, yet costs of plans change, making G less likely to be the cause of Obs whenever the cost of achieving G is smaller when Obs does not take place. We retain this definition of the test, yet we do not obtain such plans from the solution of a suitably modified version of $\hat{f}(x, u, w)$, as R&G10 do, by manipulating preconditions and effects of actions. That is not possible in *nominal models* setting, since couplings between state variables, actions and perturbation are not represented explicitly. Instead, to overcome such domain characteristic, we modify the cost function J by introducing *artificial potential fields* [95] centered on each observation o in Obs that increase or decrease costs for valid trajectories.

Let $J_\pi(x_0; G)$ be the cost functions for each of the hypothetical candidate goals $G \in \mathcal{G}$. For a given observation sequence $Obs = (o_1, \ldots, o_m)$, we define two different cost functions: J^+ that aims to *increase* the costs along the trajectories by *complying with the observation sequence Obs*; and J^- that aims to *decrease* the costs by *avoiding the observation sequence Obs*. Formally, J^+ and J^- are as follows:

$$J_\pi^+(x_0; G, Obs) = g(x_N) + \sum_{k=0}^{N-1} \left(g(x_k, \pi(x_k)) + \sum_{j=1}^{m} h(x_k, o_j) \right) \tag{4.4}$$

$$J_\pi^-(x_0; G, Obs) = g(x_N) + \sum_{k=0}^{N-1} \left(g(x_k, \pi(x_k)) - \sum_{j=1}^{m} h(x_k, o_j) \right) \tag{4.5}$$

where $h(x, o)$ is a potential field function

$$h(x, o) = 1 - exp\{-\gamma\, \ell(x - o)\} \tag{4.6}$$

where the exponent is given as some suitably defined function over the difference of vectors x and o. Recall that both x and $o \in \mathbb{R}^d$. For this thesis, we have chosen the sum *smooth abs*

functions

$$\ell(u) = \sum_i^d \sqrt{u_i^2 + p^2} + p$$

where u_i is the i-th component of the vector $x - o$ and p is a parameter we set to 1. These functions have been reported by Tassa et al. [90] to avoid numeric issues in trajectory optimization over long horizons. The potential field is used in Equation 4.4 to increase, with respect to $J_\pi(x_0; G)$, the cost of those trajectories that stay away from Obs. Conversely, in Equation 4.5 it reduces the cost for trajectories that avoid Obs. Let T^+ and T^- be sets of r best trajectories t_i^+, t_i^- for either cost function, we introduce a *cost difference function*, denoted as Δ, as follows:

$$\Delta(Obs, G) = \frac{1}{r} \sum_i^r J_{t_i^-}(x_0; G) - J_{t_i^+}(x_0; G) \tag{4.7}$$

where $J_{t_i^-}(x_0; G)$, and respectively $J_{t_i^+}(x_0; G)$, is the result of evaluating the *original* cost function $J_\pi(x_0; G)$ setting π to be the deterministic policy that follows from trajectories t_i^+ and t_i^-. We define the likelihood of Obs given G as R&G10 [78] do, as follows:

$$P(Obs \mid G) = [1 + \exp\{-\beta\ \Delta(Obs, G)\}]^{-1} \tag{4.8}$$

with the proviso that β needs to be adjusted so as to be the inverse of the order of magnitude of $\Delta(Obs, G)$. Figure 4.4 graphically illustrates the *cost difference* computation between J^+ and J^-. Note that J^+ computes trajectories that comply with the observation sequence Obs, while J^- computes trajectories that aim to avoid achieving the observed states in Obs.
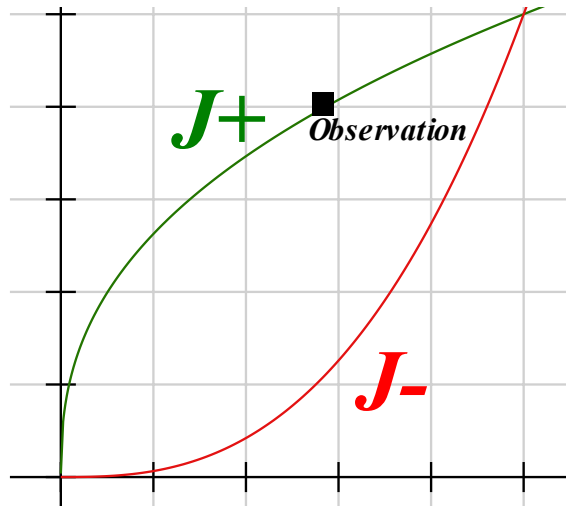
Figure 4.4: *Cost Difference* approach example.

In comparison to our previous approach (ηMIRRORING), the computational complexity of our cost difference approach is slightly different, it requires $2|\mathcal{G}|$ calls to a planner per

observation, and $2|\mathcal{G}||Obs|$ calls overall. While all methods require a number of calls linear on $|\mathcal{G}||Obs|$, evaluating Equation 4.8 tends to be more expensive, as J^+ and J^- contain several non–linear terms and their derivatives are also costlier to compute. This is relevant as most if not all of the optimization algorithms that we can use to solve Equations 2.4–2.7 rely on gradient–based techniques [16].

4.6   Experiments and Evaluation

We now present the experiments and empirical evaluations we carried out of the goal recognition approaches proposed in the previous sections. Sections 4.6.1 and 4.6.2 introduce the benchmark domains we used and describe how we generated the datasets for learning the transition function and the goal recognition tasks. In Sections 4.6.3 and 4.6.4, we report the quality of *nominal models* obtained and the performance of our goal recognition approaches over both *actual* and *nominal models* with *linear* and *non–linear* system dynamics.

4.6.1   Domains

For experiments and evaluation, we use both *linear* and *non–linear* domain models. For *linear* domains, we use three benchmark domains based on the constrained Linear–Quadratic Regulator (LQR) problem [9], a general and well–understood class of *Optimal Control* problems with countless practical applications, such as automotive active suspension control systems [37], optimal control systems for Unmanned Air Vehicle (UAV) [32], among others [11]. As for *non–linear* domains, we use two other benchmark domains based on a navigation domain that has a highly *non–linear* transition function, also used in [85, 97] to evaluate planning approaches over learned and approximate domain models.

To represent the transition between states for the LQR–based domains, we use a discrete-time *deterministic*, *linear* dynamical system

$$x_{k+1} = Ax_k + Bu_k \tag{4.9}$$

and trajectories must minimize the *quadratic* cost function

$$J = x_N^T Q x_N^T + \sum_{k=0}^{N-1} \left(x_k Q x_k^T + u_k R u_k^T \right) \tag{4.10}$$

where $Q \in \mathbb{R}^{d \times d}$ and $R \in \mathbb{R}^{p \times p}$. All matrices are set to I of appropriate dimensions, but R which is set to $10^{-2}I$. Action inputs u_k are subject to simple "box" constraints of the form $lb(u) \leq u_k \leq ub(u)$. We note that he unconstrained LQR problem has an analytical solution [10] as the cost function is globally convex and dynamics are linear. This simplifies the analysis of the behavior of optimization algorithms for training DNNs and computing trajectories. We consider two types of tasks. In the first LQR–based domain, which we call **1D–LQR–Navigation**, states $x_k \in \mathbb{R}^2$ represent the position and velocity of a particle, control inputs $u_k \in \mathbb{R}$ represent instant acceleration. Goal states require reaching a given position, yet leave terminal velocities unconstrained. The second LQR–based domain, **2D–LQR–Navigation**, has higher dimensionality as states $x_k \in \mathbb{R}^{4n}$ represent position and velocities of n vehicles on a plane, and control inputs $u_k \in \mathbb{R}^{2n}$ represent instant accelerations along the x and y axis. As in the previous domain, goal states only require reaching specific positions. For the second domain, we have two variations of this domain: 2D LQR–based navigation domain with $n == 1$, a domain with a single vehicle, denoted as **2D–LQR–Navigation–SV**, and another variation with $n == 2$, a domain with multiple-vehicles, denoted as **2D–LQR–Navigation–MV**.

As for the navigation domains with *non–linear* system dynamics, we use the same *non–linear* navigation domain defined by Say et al. in [85]. This domain consists of a navigation domain in continuous space, in which the environment has higher slippage in the center that affects directly the way the agent moves in the environment. The transition function for this *non–linear* navigation domain is defined as follows:

$$x_{k+1} = x_k + u_k \cdot 2/(1 + exp(-2 \cdot \Delta d_p)) - 0.99 \tag{4.11}$$

where states $x_k \in \mathbb{R}^D$ represent the location of an agent with D dimensions, control inputs $u_k \in \mathbb{R}^D$ represent actions to move an agent over the states axis with D dimensions, and Δd_p is the Euclidean distance between x_k and the center of the navigation environment. Based on the *non–linear* transition function defined in above in Equation 4.11, the trajectories for this

domain must minimize the following cost function

$$J = -\sum_{k-0}^{N-1} |x_G - x_k| \qquad (4.12)$$

in which the aiming of this cost function is to minimize the total Manhattan distance from the goal location x_G and the current location x_k. As the number of dimensions D is arbitrary, we consider two types of domains, domains with 2 and 3 dimensions, and we denote these two types of navigation domains as **2D–NAV** and **3D–NAV**, respectively.

4.6.2 Learning and Recognition Datasets

To build the datasets and learn the system dynamics for the domains discussed previously, we generated 500 different tasks (i.e., pairs of states x_0 and x_G) for the LQR–based domains, and 1000 different tasks for the non–linear navigation domains. We set the horizon $H = 100$ for the LQR–based domains, resulting in three different datasets with $50,000$ transitions each. As for the non–linear navigation domains (2D and 3D), we set the horizon $H = 20$, resulting in $20,000$ transitions for each domain. To generate the trajectories for each of the tasks, we first encoded the FHOC problems for each of the domains using the RDDL (Relational Dynamic Influence Diagram Language) domain description language [83]. This domain description language enabled us to use the suite of hybrid planners recently proposed by Bueno et al.. [15], which we refer to as Tf-Plan. These planners rely on state-of-the-art gradient-based optimization techniques that obtain the gradients directly from the symbolic RDDL encoding, and have been shown to offer good performance on linear and non–linear dynamics with complex non–linear cost functions. From the extracted trajectories, we build the training data for each of these domains using the states x_k, the control inputs u_k, and the resulting states x_{k+1} from x_k and u_k.

For goal recognition, we use a benchmark consisting of 30 recognition problems for **1D–LQR–Navigation**, and benchmarks with 30 problems for **2D–LQR–Navigation** setting n to 1 and 2. For the non–linear navigation domains, we use 10 recognition problems for both **2D–NAV** and **3D–NAV**. We ensure that the initial states and hypothetical candidate goals are significantly different from the ones used to learn the *nominal models*. Each recognition problem considers at most 5 goals hypotheses $\mathcal{G}$, and observations *Obs* comprise either 5 or 10 states, i.e., 5% or 10% of observability when $H = 100$, for the LQR–based domains, and

1, 2, 6, or 10 states (respectively, 5%, 10%, 30%, or 50%), when $H = 20$, for the non–linear navigation domains. To generate *Obs*, we used Tf-Plan to find a best trajectory from each initial state $\mathcal{I}$ to the hidden intended goal $G^* \in \mathcal{G}$. All states in the trajectories found have the same probability of appearing in any given *Obs*.

Figure 4.5 shows an example of a goal recognition problem for the **2D–LQR–Navigation** domain with a single vehicle. This example illustrates an example of a goal recognition problem with three candidate goals (represented by X's), an initial state (represented by a triangle), and three observed states (represented by hexagons). Consider that the intended goal is the candidate goal in the middle (Candidate Goal 1), and from this, we note that, based on the information provided by this goal recognition, it is possible to see that it is not trivial to say which goal is the intended one. This example shows the task of recognition goals over LQR–based domains in continuous space is not trivial, especially when having an approximate transition function to compute the trajectories for the possible goals considering the observations.
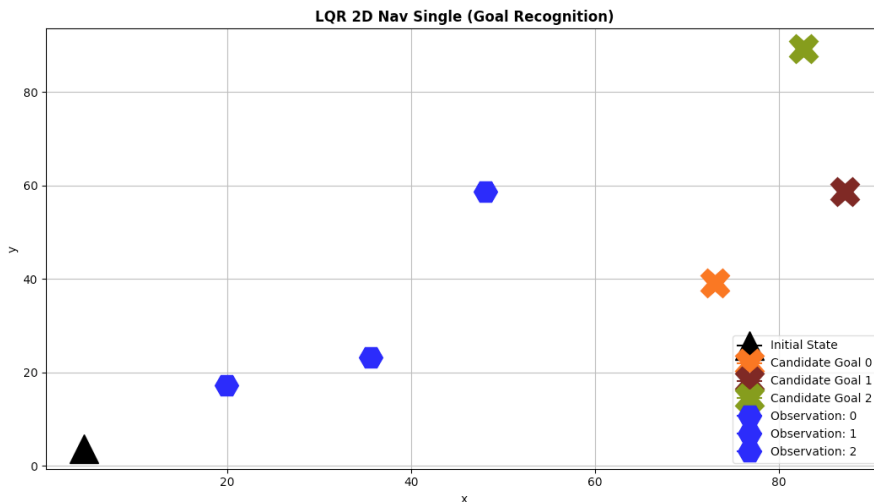
Figure 4.5: An example of a goal recognition problem for the 2D–LQR–Navigation domain with a single vehicle.

4.6.3   Learning Results

Like in Section 4.2, we learn the system dynamics of models (transition function) from data using the learning approach proposed by Say et al. in [85]. For the training stage, we configured the DNN proposed by [85] to use the same hyper-parameters to obtain *nominal models* for all domains, namely, 1 hidden layer, a batch size of 128 transitions, and we set the learning rate to 0.01 and dropout rate to 0.1. The training stage has stopped for all domains

after 300 epochs. We note that we used exactly the same DNN configuration to learn the system dynamics for all domains we described in the previous section.

Table 4.1 shows the Mean Squared Error (MSE)[1] of the best out of 10 trials after 300 epochs of training for all domains. From the errors reported in Table 4.1 we conclude that using off-the-shelf the learning approach of Say et al. [85] results in nominal models of very high quality, *as judged by the loss function* they propose. The MSE in Table 4.1 indicates we obtain accurate *nominal models* using the off-the-shelf learning approach of Say et al. [85]. Next, in Section 4.6.4, we show how our goal recognition approaches perform over the obtained *nominal models*.

Domain	MSE
1D–LQR–Navigation	$4.5 \cdot 10^{-5}$
2D–LQR–Navigation-SV	$1.7 \cdot 10^{-4}$
1D–LQR–Navigation-MV	$9.6 \cdot 10^{-6}$
2D–NAV	$6.8 \cdot 10^{-6}$
3D–NAV	$5.1 \cdot 10^{-5}$

Table 4.1: Mean Squad Error (MSE) for all domains we used.

4.6.4 Goal Recognition Results

For recognizing goals over *actual models*, we use the implementation of the Tf-Plan planner used in [15] that takes as input a domain model formalized in RDDL. For *nominal models*, we used the implementation of Tf-Plan in [97] that takes as input a domain model represented as a DNN. For both planners we set the learning rate to 0.01, batch size equals to 128, and the number of epochs to 300.

To evaluate our goal recognition approaches over both *actual* and *nominal models*, we use some of the metrics already used in the literature in goal recognition [77, 70, 67]. These metrics are the measures of True Positive Rate (TPR) and False Positive Rate (FPR). *TPR* is given by the number of true positive results (1 when G^* maximizes $P(G \mid Obs)$, 0 otherwise) over the sum of true positive results and false positive results, i.e., the number of candidate goals maximizing $P(G \mid Obs)$. A higher *TPR* indicates better performance, as it measures how often the true intended goal is calculated reliably. *FPR* is the average number of candidate goals

---

[1]The Mean Squared Error (MSE) aims to measure the average squared difference between the estimated value and the actual value.

Linear LQR–Based Domains													
				ONLINE				OFFLINE			1ST OBSERVATION		
Approach	*M*	*Obs* (%)	*N*	*Top-2*	*TPR*	*FPR*	*N*	*Top-2*	*TPR*	*FPR*	*Top-2*	*TPR*	*FPR*
ηMIRRORING	A	5	450	0.87	0.77	0.05	90	0.97	0.93	0.01	0.67	0.44	0.12
$\Delta(Obs, G)$	A	5	450	0.49	0.24	0.16	90	0.49	0.30	0.15	0.49	0.26	0.16
ηMIRRORING	A	10	900	0.90	0.78	0.05	90	0.98	0.96	0.01	0.64	0.32	0.15
$\Delta(Obs, G)$	A	10	900	0.45	0.26	0.16	90	0.44	0.28	0.15	0.47	0.28	0.15
ηMIRRORING	N	5	450	0.66	0.46	0.12	90	0.83	0.67	0.07	0.44	0.24	0.17
$\Delta(Obs, G)$	N	5	450	0.45	0.22	0.17	90	0.43	0.26	0.16	0.51	0.18	0.18
ηMIRRORING	N	10	900	0.71	0.49	0.11	90	0.87	0.72	0.06	0.41	0.26	0.16
$\Delta(Obs, G)$	N	10	900	0.45	0.26	0.16	90	0.42	0.21	0.17	0.52	0.32	0.15

Table 4.2: Experimental results of our recognition approaches over both *actual* and *nominal* models for the LQR–based domains. *M* represents the model type (A represents *actual models*, and N represents *nominal models*), % *Obs* is observation level, and *N* is the total number of observed states. Note that the average number of goal hypothesis $|\mathcal{G}|$ in the datasets is 5, and the planning horizon *H* for these datasets is 100.

$G \neq G^*$ that maximize $P(G \mid Obs)$, measuring how often goals other than the true intended one are found to be as good as or better explanation for *Obs* than G^*. We also use the *Top-k* metric, typically used in machine learning to evaluate classifiers, setting k to 2, to measure the frequency in which G^* was among the top k candidate goals as ranked by $P(G \mid Obs)$, and complements the two previous measures.

We separate the goal recognition results in two different tables: in Table 4.2, we show the results for the LQR–based domains, the domains in which the transition function is linear, whereas, in Table 4.3, we show the results for the non–linear navigation domains. In these tables, we analyze the performance of the goal recognition approaches presented in Sections 4.4 (ηMIRRORING) and 4.5 ($\Delta(Obs, G)$) in three different settings, from left to right: (1) online goal recognition (ONLINE), considering the response of the goal recognition algorithm for *each* judgment point corresponding to an observation in *Obs*; (2) offline goal recognition (OFFLINE), when we consider only the *last* judgment point (i.e., all observed states in *Obs*); and (3) considering only the *first* judgment point (1ST OBSERVATION), i.e., $o_1 \in Obs$. Note that, in these tables, we aggregate and summarize the average results for all evaluated domains, and in APPENDIX C, we provide extensive experimental evaluation for all domain models separately, for both linear and non–linear domains.

As for the results for the linear (LQR–based) domains in Table 4.2, we note that under the parameters used for the planners, ηMIRRORING clearly dominates $\Delta(Obs, G)$ in all settings by a wide margin, except for the results when considering the first observation over *nominal*

models. Seeking an explanation for the poor performance of $\Delta(Obs, G)$, we dug deeper into the experimental data in order to find how often ηMIRRORING outperformed $\Delta(Obs, G)$ and vice versa. Interestingly, we found that ηMIRRORING outperforms $\Delta(Obs, G)$, according to the *Top–2* measure, in 37% of the judgment points considered, $\Delta(Obs, G)$ is superior in 8.9% of the cases and both approaches are in agreement and correct in 42.1% of cases. This suggests that $\Delta(Obs, G)$ could be sensitive to one of the parameters used to calculate the trajectories. Over this ablation study, we detected that the number of epochs is the key parameter, as it directly affects how far from optimal are the trajectories found. We also observed that varying the number of epochs had counter-intuitive results, as the approximations to the optimal values of J^+ and J^- do not get better or worse in a linear fashion. Instead, we often observed costs improve (or worsen) for either cost functions at different rates, sometimes changing the sign of $\Delta(Obs, G)$. To analyze the impact of the number of epochs over the $\Delta(Obs, G)$, we ran and tested the $\Delta(Obs, G)$ over a limited number of instances, setting the number of epochs to $3,000$, and we observed a significant improvement which brought it to be in agreement with the performance of ηMIRRORING if not sometimes superior. Of course, this entailed an increase of run times by roughly an order of magnitude. Thus, this leads us to conclude that the relatively good results of $\Delta(Obs, G)$ in Table 4.2 are due to the fact that J^+ and J^- are closer to the convex ideal in Equation 4.10, as they include less non–linear terms $h(x, o)$, so TF-PLAN is less likely to get trapped in a local minima with adverse results for recognition accuracy early on.

In comparison to the results for the linear LQR–based domain models (Table 4.2), we can see that the results for the non–linear domains are better for both our recognition approaches over all settings, as shown in Table 4.3. It is also possible to see that, again, ηMIRRORING dominates $\Delta(Obs, G)$ in all settings, but not as much as it is for the linear LQR–based domains. For non–linear domains, ηMIRRORING is outperforms $\Delta(Obs, G)$, when using the *Top–2* measure, in 8.8% of the judgment points considered. Moreover, $\Delta(Obs, G)$ outperforms ηMIRRORINGin 3.5% of the cases, and both recognition approaches are in agreement and correct in 30.1% of cases. An interesting aspect regarding the results for non–linear domains is about performance compared to the linear LQR–based domain models. We note that the main difference between the recognition datasets for linear and non–linear domains is the length of the planning horizon H, which is 100 for the linear LQR–based domains, and 20 for the non–linear navigation domains. This may lead us to conclude that the length of the horizon affects the planning process when extracting trajectories, by accumulating and propagating error along trajectories, especially for nominal models.

			Non–Linear Navigation Domains										
				Online				Offline			1st Observation		
Approach	*M*	*Obs (%)*	*N*	*Top-2*	*TPR*	*FPR*	*N*	*Top-2*	*TPR*	*FPR*	*Top-2*	*TPR*	*FPR*
ηMirroring	A	5	20	0.85	0.85	0.04	20	0.85	0.85	0.04	0.85	0.85	0.04
$\Delta(Obs, G)$	A	5	20	0.80	0.65	0.09	20	0.80	0.65	0.09	0.80	0.65	0.09
ηMirroring	A	10	40	0.90	0.78	0.06	20	1.00	0.90	0.03	0.80	0.65	0.09
$\Delta(Obs, G)$	A	10	40	0.55	0.38	0.16	20	0.55	0.30	0.17	0.55	0.45	0.14
ηMirroring	A	30	120	0.89	0.75	0.06	20	1.00	1.00	0.00	0.75	0.55	0.11
$\Delta(Obs, G)$	A	30	120	0.55	0.33	0.17	20	0.55	0.35	0.16	0.55	0.35	0.16
ηMirroring	A	50	200	0.88	0.73	0.07	20	1.00	1.00	0.00	0.75	0.35	0.16
$\Delta(Obs, G)$	A	50	200	0.57	0.32	0.17	20	0.85	0.40	0.15	0.50	0.25	0.19
ηMirroring	N	5	20	0.85	0.70	0.07	20	0.85	0.70	0.07	0.85	0.70	0.07
$\Delta(Obs, G)$	N	5	20	0.75	0.40	0.15	20	0.75	0.40	0.15	0.75	0.40	0.15
ηMirroring	N	10	40	0.72	0.55	0.11	20	0.80	0.65	0.09	0.65	0.45	0.14
$\Delta(Obs, G)$	N	10	40	0.55	0.30	0.17	20	0.60	0.40	0.15	0.50	0.20	0.20
ηMirroring	N	30	120	0.76	0.61	0.10	20	1.00	1.00	0.00	0.50	0.25	0.19
$\Delta(Obs, G)$	N	30	120	0.59	0.28	0.18	20	0.75	0.40	0.15	0.50	0.15	0.21
ηMirroring	N	50	200	0.81	0.65	0.09	20	1.00	1.00	0.00	0.50	0.25	0.19
$\Delta(Obs, G)$	N	50	200	0.54	0.27	0.18	20	0.70	0.55	0.11	0.40	0.15	0.21

Table 4.3: Experimental results of our recognition approaches over both *actual* and *nominal* models for the non–linear navigation domains (2D–NAV and 3D–NAV). *M* represents the model type (A represents *actual models*, and N represents *nominal models*), % *Obs* is observation level, and N is the total number of observed states. Note that the average number of goal hypothesis $|\mathcal{G}|$ in these datasets is 4, and the planning horizon H for these datasets is 20.

With respect to recognition time, the average time per goal recognition problem for ηMirroring over the linear LQR–based datasets is $\approx 1,100$ seconds, whereas for $\Delta(Obs, G)$ is $\approx 1,600$ seconds. As for the recognition time of our approaches when dealing with non–linear domains, the average time per problem is $\approx 1,900$ seconds for ηMirroring, and $\approx 2,500$ seconds for $\Delta(Obs, G)$. Note that the (linear and non–linear) domains and problems we used in our experiments are non-trivial real-world domains, and all of them have continuous state–space and actions, in which the planning process usually takes substantial time for extracting optimal trajectories.

4.7   Chapter Remarks

Model–based goal and plan recognition is a real-world, non–trivial and challenging application of causal reasoning, and, in this chapter, we adapt past approaches to model–based goal recognition as different implementations of Halpern's *but-for* test of sufficient causality. We also show that learning techniques can be used to generate predictions which are good

enough to enable the generation of meaningful counterfactuals and by extension "true" causal reasoning [64].

Future work on goal recognition over *nominal models* should investigate three key questions. First, to what extent the proposed recognition approaches can handle increasing variance for the random variable w. Second, we need to determine whether it is possible to modify the loss function used for training the nominal models in a way that takes into account the accumulated error along trajectories, rather than just the errors in predicting the next state. Last, as discussed in Section 4.6.4, cost–based goal recognition is very sensitive to planners converging to unhelpful local minima, which seems to be an inherent characteristic of stochastic optimization algorithms. These recognition approaches could also be evaluated using planners that rely on Differential Dynamic Programming (DDP) [56, 98], which *may* converge faster to better local minima of the cost function.

In this chapter, we developed novel approaches to goal recognition that expand the applicability of model–based goal and plan recognition by replacing carefully engineered models for carefully curated datasets. The approaches developed in this chapter are also examples of how to exploit latent synergies between *Planning*, *Optimal Control*, *Optimization*, and *Machine Learning*, as we integrate algorithms, techniques, and concepts to address in novel ways a high-level, transversal problem relevant to many fields in *Artificial Intelligence*.

5.   RELATED WORK

In this chapter, we survey and review the most significant work on *Goal and Plan Recognition* and *Planning* that are closely related to the contributions presented in this thesis. In Section 5.1, we present model–based approaches to goal and plan recognition that directly rely on *Planning* techniques. After, in Section 5.2, we describe the approaches in the literature that deal with incomplete domain information for recognizing goals and plans. Finally, in Section 5.3, we describe existing work on planning over *imperfect domain models*, namely, planning approaches that deal with either incomplete discrete domain models, or approximate hybrid (continuous and discrete) domain models.

5.1   Goal and Plan Recognition as Planning

Over the past years, the task of goal and plan recognition as planning has received much attention in the *Automated Planning* community, resulting in several remarkable contributions in recent years. One of the first planning–based approach for recognizing goals and plans was developed by Hong [36]. Hong develops an approach that extends the concept of planning graph [12], proposing a similar structure that represents every possible path (e.g., state transitions that connect facts and actions) from an initial state to a goal state, and calling this structure a goal graph. As actions are observed during a plan execution, a goal graph is constructed, in which facts that represent recognized goals are linked to a goal level. This work is one the first work to address the task of goal recognition without using plan–libraries, showing that it is possible to be fast and accurate for recognizing goals without explicitly defining the plans that achieve the goals.

Later, Ramírez and Geffner [77] introduce the problem of *Plan Recognition as Planning*, by using planning domain models to describe the agents' behavior and planning techniques to perform the recognition task. For recognizing goals and plans, Ramírez and Geffner use modified (optimal and sub–optimal) planning algorithms to determine the distance to every goal in a set of candidate goals given a sequence of observations. In this work, Ramírez and Geffner develop two approaches, specifically, they consider optimal or sub–optimal plans, in which goals that have become impossible being removed from the set of candidate goals. Once candidate goals have been eliminated, they are never reconsidered during the recognition process. Ramírez and Geffner [77] also work with an assumption of partial observability in that

only a sub-sequence of the plan is available as evidence to the recognizer. In their subsequent work, Ramírez and Geffner [78] develop a robust probabilistic framework for goal and plan recognition by using off-the-shelf planners, providing a posterior probability distribution over goals, given an observation sequence as evidence.

In [63], Pattison and Long propose AUTOGRAPH (AUTOmatic Goal Recognition with A Planning Heuristic), a probabilistic heuristic-based goal recognition approach over planning domains. Much like the work of Ramírez and Geffner [77], AUTOGRAPH uses heuristic estimation and domain analysis to determine which goal(s) a plan execution of an observed agent is pursuing. However, unlike most work on goal and plan recognition as planning, the set of candidate goals is not given as part of the recognition problem, the intended goal is inferred based on a planning domain model, an initial state, and a sequence of observations.

As one of the very first to address the task of *Plan Recognition as Planning* with multiple agents, in [104], Zhuo et al. develop a recognition approach in which the team behavior model is defined as a planning domain definition (i.e., every agent behavior is based on the planning domain definition), and the task of plan recognition analyzes a partial team observation trace for recognizing team plans. To perform the recognition process, Zhuo et al. first translate the multi-agent plan recognition as a satisfiability problem, and then solve the recognition problem by using a weighted MAX-SAT solver. Zhuo et al. also show a comparison between their previous plan–library based approach [102] and the planning domain definition based approach.

Dealing explicitly with ambiguity in goal and plan recognition is a very complex task. In [40], Keren et al. develop an alternate view of the goal recognition problem, and rather than developing new goal recognition algorithms, they develop novel techniques that modify the domain model in order to facilitate the goal recognition process. Specifically, the approaches of Keren et al. [40, 41, 42] aim to attempt to reduce the number of non-unique plans for each candidate goal in a set of candidate goals, and thus, simplifying the process of goal recognition by redesigning the planning domain model.

E.-Martín et al. [19] propose a planning-based goal recognition approach that propagates cost and interaction information in a plan graph, and uses this information to estimate goal probabilities over the set of candidate goals and the observation sequence. We note that the approach of E.-Martín et al. [19] is the first one in the literature that obviates calling a planner to perform the recognition task, resulting in a very fast approach to goal recognition.

Sohrabi et al. [87] extend the probabilistic framework of Ramírez and Geffner [78], and developed a novel probabilistic recognition approach that deals explicitly with unreliable and spurious observations (i.e., noisy or missing observations), and recognizes both goals and plans. The probabilistic approaches of Sohrabi et al. [87] use multiple high-quality plans (using a planner that generates multiple plans with high quality) to produce a probabilistic distribution over the goals. In this paper, the authors show that, for some domains, the use of multiple high-quality plans along with this novel probabilistic framework yields better results than using only one plan [78].

Vered, Kaminka, and Biham et al. [92] introduce the concept of *Mirroring* to develop an online goal recognition approach for continuous domain models. Based on this work, Vered et al. [93] develop an online goal recognition approach that combines the concept of *Mirroring* and landmarks, showing this combination can improve not only the recognition time, but also the accuracy for recognizing goals in the online fashion. In [39], Kaminka et al. propose a new probabilistic framework for plan recognition approach over both continuous and discrete domains, in which the core of this framework is the concept of *Mirroring*.

Masters and Sardiña [48, 49] propose a fast and accurate goal recognition approach that works strictly in the context of path-planning, providing a novel probabilistic framework for goal recognition in path planning, which is basically a revised and improved version of the probabilistic framework of Ramírez and Geffner [78]. This novel probabilistic framework for path-planning shows that it is possible to compute the probability distribution over the goals much simpler and faster than the one proposed by Ramírez and Geffner [78], considering only a single observation, namely the current state. In their most recent work, Masters and Sardiña [50] improve their previous probabilistic approach to deal with both rational and irrational agent behavior during the recognition process.

In previous work [65, 70], we develop landmark–based approaches for goal recognition as planning. Such approaches are recognition heuristics that strictly rely on the concept of landmarks. Their first heuristic approach performs the recognition task by computing the ratio between the number of achieved landmarks and the total number of landmarks for a given candidate goal, and then, the candidate goal (s) with the highest heuristic value is (are) considered as the most likely intended one. The second heuristic approach uses the concept of *landmark uniqueness value*, representing the information value of the landmark for a particular candidate goal when compared to landmarks for all candidate goals. Thus, the heuristic estimation provided by this second heuristic is the ratio between the sum of the uniqueness value of the

achieved landmarks and the sum of the uniqueness value of all landmarks of a candidate goal. Most recently, in their extended work, in [72], they show through several experiments that these landmark–based heuristics are the fastest ones in the literature.

Freedman et al. [22] proposed an approach to perform probabilistic plan recognition along the lines of the work of Ramírez and Geffner [78], in which, instead of calling a full-fledged planner for each candidate goal, it takes advantage of a multiple-goal heuristic search algorithm [17] to search for all goals simultaneously, avoiding repeatedly expanding the same nodes in the search tree. This approach has not been implemented and evaluated yet, the authors have provided only theoretical concepts regarding this approach.

Unlike the heuristic approaches we developed in Chapter 3, most recent planning-based recognition approaches [77, 78, 87, 92] use a planner to recognize goals and plans from observations, calling a planner at least $2 \times \mathcal{G}$ times during the recognition process. Conversely, E.-Martín et al. [19] and Pereira et al. [70] are similar to our heuristics approaches because these approaches avoid the use of automated planners during the goal and plan recognition process, and use only planning information extracted from planning instances, i.e., planning-graphs and landmarks, respectively. Keren et al. [40] developed an approach that assumes planning domain models are not fixed, and it changes (re-designs) the domain definition to facilitate the task of goal recognition in planning domain models. However, these approaches differ from ours because they only deal with complete (even if modified) domain models, and most of them transform/compile the goal/plan recognition problem into a planning problem to be solved by a planner. Such a transformation or compilation process may not necessarily work with incomplete STRIPS domain models, given the very large number of potential models. We note that the approach of E.-Martín et al. [19] could work in incomplete domain models with some adaptations (e.g., by ignoring all *possible* preconditions and effects), though this approach would likely be less accurate than our approaches (as shown in [72] for complete and correct domain models) because it does not deal intentionally with possible preconditions and effects, while our recognition approaches do.

5.2   Planning over Imperfect Domain Models

There has been comparatively little research in the *Automated Planning* literature to deal explicitly with inaccurate and imperfect domains models. One of the first planning approaches to address incomplete information in discrete domain models is the work of Garland and Lesh [23]. In this work, Garland and Lesh [23] develop a planning approach for incomplete domain models, allowing the use of annotations (possible preconditions and effects) to specify incomplete actions in the domain description. Their planning approach analyzes a set of extracted plans that identifies critical faults (facts that may cause plan failure) in these action sequences, returning the plan with the best quality (i.e., the plan with the minimal number of critical faults). Weber and Brycen [96] use the same annotations for incomplete domains, and develop a planner called DEFAULT, which aims to search for plans by minimizing their risks to fail for achieving goals. DEFAULT uses a heuristic approach based on the Fast-Forward (FF) heuristic [34], breaking ties using a novel heuristic that counts failure models, called PRIME IMPLICANT heuristic. Most recently, Nguyen et al. [60] develop two approaches for planning in incomplete domain models, the PISA and $\mathcal{C}$PISA planners [61]. PISA is a planner that uses a stochastic local search to synthesize robust plans in incomplete planning domains. $\mathcal{C}$PISA extends the techniques from PISA incorporating Bayesian learning to enhance the planning process. PISA and $\mathcal{C}$PISA outperform DEFAULT for planning in most incomplete domain models. The incomplete domain formalism we use for the task of goal recognition over incomplete domains in (as presented in Chapter 3) is based on the work presented above [23, 96, 61, 60].

Recent *Deep Learning* techniques have shown to be very effective to learn linear and non-linear transition functions from data. Say et al. [85] use state-of-the-art deep learning techniques to approximate the transition function of hybrid (mixed discrete and continuous) domain models, based on datasets that contain plan traces represented as state transitions. For planning, Say et al. developed a Mixed-Integer Linear Program (MILP) based planner that works in two states: (1) it encodes the learned transition function and a hybrid domain model into a MILP Program; and (2) given this MILP encoding, Say et al. use an off-the-shelf MILP solver to find plans for a given planning horizon. Subsequently, Wu, Say, and Sanner [97] use the same approach to approximate the transition function of hybrid domain models, but unlike the work of Say et al. [85], they use pure learning techniques to develop their planning approach. Namely, they develop a planning approach based on Tensorflow [1] and a gradient descent

optimization (RMSProp[1]). Over extensive experimental results, they show that the resulting learning approach is very competitive in comparison to the MILP based planner on several linear and non-linear hybrid planning domains. Recently, in [84], Say and Sanner develop two alternative approaches to the ones presented in [85, 97]. Unlike the previous approaches, the recent approaches of Say and Sanner [84] use modern *Machine Learning* approaches to learn the transition function of hybrid domain models, whereas for planning, they develop two approaches: the first one compiles the learned transition function and a hybrid domain model into a Boolean Satisfiability problem, and use SAT solver to find plans for a given horizon, while the second one compiles the problem into a Binary Linear Programming (BLP) formulation, and then use a BLP solver for the planning process.

As we presented in Chapter 4, the core of our goal recognition approaches over *nominal models* is inspired by the work of Say et al. [85] and Wu, Say, and Sanner [97]. To learn and approximate transition functions from datasets, we use the learning approach of Say et al. [85], and for planning over nominal models, we make use of the Tensorflow planner developed by Wu, Say, and Sanner in [97].

5.3   Plan Recognition with Incomplete Domain Information

To the best of our knowledge, the earlier work on goal and plan recognition that deal explicitly with *incomplete domain models* are that of Lee and McCartney [46] and Kerkez and Cox [43], and most recently, the work of Zhuo [101]. The main characteristic that these approaches [46, 43, 101] have in common with our approaches to goal recognition over incomplete domains (Chapter 3), is the use of incomplete information in the domain model description.

Lee and McCartney [46] developed a plan recognition approach that uses stochastic models (Hidden Markov Models) to model possible ambiguities in the agent behavior, and learning techniques to learn actions and properties of the model based on an incomplete agent behavior model from partial observation, which are stored as a history of interactions in a dataset. To describe the agent behavior model, the authors use a graph structure similar to a plan–library, but with incomplete information, and in this incomplete behavior model, such incomplete information represents the set of unknown properties and actions of the agent behavior model.

---

[1]Developed by Geoff Hinton (in Lecture 6 of his Coursera Class), RMSprop is an unpublished optimization algorithm designed for Neural Networks.

Unlike the approach of Lee and McCartney [46] that uses learning techniques to fill the incomplete part of the domain model, the plan recognition approach of Kerkez and Cox [43] takes as input an incomplete plan–library and deals with incomplete domain information by using an automated planner. More specifically, the approach of Kerkez and Cox uses a planner to fill and complete an incomplete plan–library from the observations, and then recognizes the observed agent's goal using a mapping technique, matching the resulting plan–library with the observations.

Zhuo [101] develops a multi-agent plan recognition approach that uses as domain knowledge the combination of an incomplete action description model and a set of incomplete team plans (i.e., a set of possible team plans that the agents can perform to achieve their goals). For recognizing multi-agent team plans, the approach works in two stages: (1) the approach transforms the multi-agent recognition problem as a satisfiability problem, encoding the problem by using soft and hard constraints; and (2) based on the encoded constraints, the approach then solves the recognition problem by using a weighted MAX-SAT solver.

We argue that the plan recognition approaches [46, 43, 101] described above are quite different from the heuristic recognition approaches over incomplete domains we developed and presented in Chapter 3 in several key aspects, and we single out these differences as follows. Firstly, these recognition approaches in [46, 43, 101] use incomplete plan–libraries or incomplete team plans to represent the agent behavior model, whereas our heuristic approaches only use incomplete planning domain models. Plan–libraries are usually encoded manually and laborious to model, requiring a description of a set of plans in order to know how to achieve the set of possible goals. Even using incomplete information, the task of modeling plan–libraries can be quite complex, requiring much design effort and domain knowledge. Moreover, plans that are not defined in the plan–libraries cannot be recognized during the recognition process. We note the task of modeling incomplete planning domain models requires much less effort, requiring only a description of the set of predicates (properties) and actions of the environment. Secondly, another key difference is that we use no learning approach to learn or fill in the incomplete part of the domain model, and deal explicitly with the incomplete domain information. Finally, we use no planner or any other kind of solver for recognizing goals over incomplete domain information and only use the information provided by the landmark extraction process.

We have not provided any comparison against these approaches because the formalisms they use for incomplete domains are incompatible with the formalism we use in our work. However, a comparison against the work of Zhuo [101] would be possible if we modify his

work to cope only with a single agent, generate a set of incomplete plans for a single agent for achieving the possible candidate goals, and adapt our work to his formalism for incomplete domain models. We argue that, even with such adaptations and modifications, the comparison would not be fair, since Zhuo's approach assumes plans both in the plan–library and in the observations have a fixed size and contain information about exactly what are times of the missing observations, which our work does not assume.

5.4   Chapter Remarks

We conclude this chapter by noting that, we have not surveyed all approaches to goal and plan recognition in the literature, and instead, we chose to focus on presenting the most relevant approaches to *Plan Recognition as Planning*. The literature of goal and plan recognition is vast, and contains several types of significant approaches that are not directly related to *Automated Planning*, for instance, goal and plan recognition approaches that rely on plan–libraries [6, 55, 54] and context-free grammars [27]. Thus, we state that the main focus of this chapter is presenting existing work that is based on planning techniques and incomplete domain information.

6.   CONCLUSIONS

In this thesis, we introduced new formalizations for goal recognition problems that allow for *imperfections* over two distinct types of domains models, i.e., *incomplete discrete domain models* that have *possible*, rather than *known*, preconditions and effects in action descriptions, and *approximate continuous domain models*, where the transition function is *approximate* and not well-defined. We developed novel goal recognition approaches that can cope with these two types of imperfect domain models, and we have empirically shown that such approaches are accurate when dealing with imperfect domains in several recognition settings. In Section 6.1, we summarize and discuss the main contributions of this thesis, and after that, in Sections 6.2 and 6.3, we discuss, respectively, the open issues and limitations of our approaches, as well as the avenues that we can for improving our recognition approaches.

6.1   Contributions

We now outline the two main contributions of this thesis, as follows.

1. Our first contribution, presented in Chapter 3, is related to the task of goal recognition over *incomplete discrete domain models*, and resulting in the following the specific contributions.

   - *A new problem formalization for goal recognition over incomplete domain models* (Section 3.1), combining the standard formalization of goal recognition of Ramírez and Geffner [77, 78] with the formalization of incomplete domain models introduced by Nguyen et al. [61, 60]. This new problem formalization allows the use of incomplete domain models for recognizing goals, relaxing the need for complete and correct discrete domain models;

   - *A novel landmark extraction algorithm that deals with incomplete domain models* (Section 3.2), adapted from [35], enabling us to use new notions of landmarks from incomplete planning instance, and therefore, obtaining more information for recognizing goals over incomplete domains;

   - *New notions of landmarks for incomplete domain models* (Section 3.2, formally defined in Definitions 16, 17, and 18), namely, *definite*, *possible*, and *overlooked landmarks*, that we use to develop enhanced recognition heuristics [70]; and

- *Enhanced landmark–based heuristics for goal recognition over incomplete domain models* (Sections 3.3.1 and 3.3.2), developed based on the new notions of landmarks over incomplete domain models. Experiments over thousands of goal recognition problems (in fifteen incomplete planning domain models) show that our enhanced recognition approaches are fast and accurate when dealing with incomplete domains at all variations of observability and percentage of domain incompleteness.

2. The second contribution of this thesis is about the problem of recognizing goals over *approximate continuous domain models*, presented in Chapter 4. We now enumerate our specific contributions towards solving this problem.

- *A new problem formalization for goal recognition over nominal models* (Section 4.1), that extends the formalization of Ramírez and Geffner [77, 78] by reasoning about the agent behavior using FHOC problems, in which the transition function is not well-defined but approximate. We note that this new formalization relaxes the need for using known and well-defined transition functions, allowing the representation of transition functions as "black boxes", which can be learned from data using learning techniques [85];

- *A goal recognition approach based on the concept of Mirroring* (Section 4.4), that adopts the probabilistic framework of Kaminka et al. [39] and improves the efficiency of the original *Mirroring* approach [92]. In Section 4.6, we show that this approach is accurate when dealing with *linear* and *non-linear* domain models at very low levels of observability;

- *A goal recognition approach based on cost-differences* (Section 4.5), that follows the well-known probabilistic approach of Ramírez and Geffner in [78]. Our recognition approach uses the concept of artificial potential fields, like [78], but computes the cost-difference based on the values of modified cost functions. Experiments and evaluation showed that the latter approach is not as accurate as the former, possibly because the planner we use converges to unhelpful local minima when extracting the trajectories.

6.2   Open Issues and Limitations

The contributions of this thesis are based on over two distinct types of domains models and techniques, and consequently, the issues and limitations of such contributions are not related. In this section, we discuss the issues and limitations of our approaches to goal recognition over *imperfect domain models*, and identify possible approaches to overcome such issues and limitations in future work.

As our ablation study in Section 3.4.5 shows, the main limitation of our enhanced heuristic approaches to goal recognition over *incomplete domain models* is dealing with low observability. Our approaches are sensitive to the amount of landmark information, and as a result, they are not accurate at low levels of observability when increasing the percentage of domain incompleteness, i.e., for 10% and 30% of observability when the percentage of domain incompleteness is more than 60%. he set of *possible preconditions* may provide a new source of information during the landmark extraction, since our landmark extraction algorithm only explores the set of *possible add effects* by ignoring the possible preconditions to build an ORPG. By doing so, one can potentially obtain more landmarks, and as a result, have more information to use in our heuristic approaches.

The main limitation of our approaches to goal recognition over *nominal models* is that they rely on good trajectories to be accurate. Therefore, such recognition approaches are sensitive to the trajectories provided by the planner. We see in Section 4.6 that, the results of one of our recognition approaches (Section 4.5) have been directly affected by the gradient-based optimization planner [15, 97] we used in our experiments. We carefully analyzed the extracted trajectories for this approach, and we identified that for most problems this planner is converging to local minima. A possible solution to this would be using a different planner that relies on other optimization techniques, for instance, a planner that exploits Dynamic Programming (DDP) [56, 98].

6.3   Future Work

Besides addressing the issues and limitations we pointed out above, as future work, we aim to explore multiple avenues to extend the goal recognition approaches we presented in this thesis. With respect to our work on goal recognition over *incomplete discrete domain models*, we intend to explore two potential ideas to extend this work. First, we intend to use a propagated RPG to reason about *impossible incomplete domain models*, much like in [96], in which the authors use such information to build a planning heuristic for planning over incomplete domain models. Second, we aim to explore recent work that could be used as part of a complete methodology to develop domains includes an approach to acquire and infer information from domains with incomplete information based plan traces. In this work, Zhuo et al. [103] developed an approach to refine incomplete domain models based on plan traces.

In order to improve our work on goal recognition over *nominal models*, we intend to investigate modern probabilistic frameworks for goal recognition from the literature that cope with Euclidean space, such as the one introduced by Masters and Sardiña in [48, 49]. An interesting extension for this work would be addressing another imperfect aspect in this type of model, such as *approximate cost functions*. This involves learning the *cost function* from data, much like we did to learn the transition function (Section 4.2), but using a different learning technique, such as Linear Regression [99]. We have conducted some preliminary tests in this regard, by learning the cost function from data, and using it along with *nominal models*. Specifically, we modified the implementation of TF-PLAN in [97] to extract trajectories over models in which both the transition and cost functions are approximate. Our preliminary tests showed that for most problems the planner is overestimating the states in the trajectories. To overcome this issue, we intend to use the learned approximate functions along with another planner, and then evaluate our approaches over this extended setting, in which both the transition and cost functions are approximate.

BIBLIOGRAPHY

[1] Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G. S.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Goodfellow, I. J.; Harp, A.; Irving, G.; Isard, M.; Jia, Y.; Józefowicz, R.; Kaiser, L.; Kudlur, M.; Levenberg, J.; Mané, D.; Monga, R.; Moore, S.; Murray, D. G.; Olah, C.; Schuster, M.; Shlens, J.; Steiner, B.; Sutskever, I.; Talwar, K.; Tucker, P. A.; Vanhoucke, V.; Vasudevan, V.; Viégas, F. B.; Vinyals, O.; Warden, P.; Wattenberg, M.; Wicke, M.; Yu, Y.; Zheng, X. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems", *Computing Research Repository (CoRR)*, vol. abs/1603.04467, Mar 2016, pp. 1–19.

[2] Amado, L.; Pereira, R. F.; Aires, J. P.; Magnaguagno, M.; Granada, R.; Licks, G. P.; Meneguzzi, F. "Latrec: Recognizing goals in latent space". In: Proceedings of the System Demonstrations and Exhibits at the International Conference on Automated Planning and Scheduling (ICAPS), 2019, pp. 1–2.

[3] Amado, L.; Pereira, R. F.; Aires, J. P.; Magnaguagno, M.; Granada, R.; Meneguzzi, F. "Goal recognition in latent space". In: Proceedings of the International Joint Conference on Neural Networks (IJCNN), 2018, pp. 1–8.

[4] Armentano, M. G.; Amandi, A. "Plan recognition for interface agents", *Artificial Intelligence Review*, vol. 28–2, Aug 2007, pp. 131–162.

[5] Asai, M.; Fukunaga, A. "Classical Planning in Deep Latent Space: Bridging the Subsymbolic-Symbolic Boundary". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2018, pp. 6094–6101.

[6] Avrahami-Zilberbrand, D.; Kaminka, G. A. "Fast and Complete Symbolic Plan Recognition". In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2005, pp. 653–658.

[7] Avrahami-Zilberbrand, D.; Kaminka, G. A. "Incorporating observer biases in keyhole plan recognition (efficiently!)". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2007, pp. 944–949.

[8] Baker, C. L.; Joshua B. Tenenbaum, J. B.; Saxe, R. "Action understanding as inverse planning", *Cognition*, vol. 113–3, Jul 2009, pp. 329–349.

[9] Bemporad, A.; Morari, M.; Dua, V.; N. Pistikopoulos, E. "The explicit linear quadratic regulator for constrained systems", *Automatica*, vol. 38, Jan 2002, pp. 3–20.

[10] Bertsekas, D. P. "Dynamic Programming and Optimal Control". Athena Scientific, 2017, 4th ed., 520p.

[11] Beteto, M. A.; Assunção, E.; Teixeira, M. C.; Silva, E. R.; Buzachero, L. F.; Caun, R. P. "New Design of Robust LQR-State Derivative Controllers via LMIs", *International Federation of Automatic Control*, vol. 51–25, Nov 2018, pp. 422–427.

[12] Blum, A. L.; Furst, M. L. "Fast Planning Through Planning Graph Analysis", *Journal of Artificial Intelligence Research*, vol. 90, Feb 1997, pp. 281–300.

[13] Borrelli, F.; Bemporad, A.; Morari, M. "Predictive control for linear and hybrid systems". Cambridge University Press, 2017, 1st ed., 440p.

[14] Bryce, D.; Kambhampati, S. "A Tutorial on Planning Graph Based Reachability Heuristics", *AI Magazine*, vol. 28–1, Mar 2007, pp. 47–83.

[15] Bueno, T. P.; Barros, L.; Maua, D. D.; Sanner, S. "Deep reactive policies for planning in stochastic nonlinear domains". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2019, pp. 7530–7537.

[16] Calafiore, G. C.; El-Ghaoui, L. "Optimization Models". Cambridge University Press, 2014, 1st ed., 650p.

[17] Davidov, D.; Markovitch, S. "Multiple-goal heuristic search", *Journal of Artificial Intelligence Research*, vol. 26, Aug 2006, pp. 417–451.

[18] Dennett, D. "Intentional systems in cognitive ethology: The "panglossian paradigm defended"", *Behavioral and Brain Sciences*, vol. 6, Sep 1983, pp. 343–390.

[19] E.-Martín, Y.; R.-Moreno, M. D.; Smith, D. E. "A Fast Goal Recognition Technique Based on Interaction Estimates". In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2015, pp. 761–768.

[20] Fernández-González, E.; Williams, B. C.; Karpas, E. "Scottyactivity: Mixed discrete-continuous planning with convex optimization", *Journal of Artificial Intelligence Research*, vol. 62, Jul 2018, pp. 579–664.

[21] Fikes, R. E.; Nilsson, N. J. "STRIPS: A new approach to the application of theorem proving to problem solving", *Artificial Intelligence*, vol. 2–3, Sep 1971, pp. 189–208.

[22] Freedman, R. G.; Fung, Y. R.; Ganchin, R.; Zilberstein, S. "Towards quicker probabilistic recognition with multiple goal heuristic search". In: The Workshop on Plan, Activity, and Intent Recognition (PAIR) at the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2018, pp. 1–8.

[23] Garland, A.; Lesh, N. "Plan evaluation with incomplete action descriptions". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2002, pp. 461–467.

[24] Geffner, H.; Bonet, B. "A Concise Introduction to Models and Methods for Automated Planning". Morgan & Claypool, 2013, 1st ed., 141p.

[25] Geib, C. W. "Problems with Intent Recognition for Elder Care". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2002, pp. 13–17.

[26] Geib, C. W.; Goldman, R. P. "Plan Recognition in Intrusion Detection Systems". In: Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX), 2001, pp. 46–55.

[27] Geib, C. W.; Goldman, R. P. "A Probabilistic Plan Recognition Algorithm Based on Plan Tree Grammars", *Artificial Intelligence*, vol. 173–11, Jul 2009, pp. 1101–1132.

[28] Ghallab, M.; Nau, D. S.; Traverso, P. "Automated Planning - Theory and Practice." Elsevier, 2004, 1st ed., 635p.

[29] Ghallab, M.; Nau, D. S.; Traverso, P. "Automated Planning and Acting". Elsevier, 2016, 1st ed., 368p.

[30] Goodfellow, I.; Bengio, Y.; Courville, A. "Deep Learning". MIT Press, 2016, 1st ed., 775p.

[31] Granada, R.; Pereira, R. F.; Monteiro, J.; Barros, R.; Ruiz, D.; Meneguzzi, F. "Hybrid Activity and Plan Recognition for Video Streams". In: The Workshop on Plan, Activity, and Intent Recognition (PAIR) at the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2017, pp. 1–8.

[32] Hajiyev, C. "LQR Controller with Kalman Estimator Applied to UAV Longitudinal Dynamics", *Positioning*, vol. 04, Jan 2013, pp. 36–41.

[33] Halpern, J. Y. "Actual Causality". The MIT Press, 2016, 1st ed., 229p.

[34] Hoffmann, J.; Nebel, B. "The FF Planning System: Fast Plan Generation Through Heuristic Search", *Journal of Artificial Intelligence Research*, vol. 14, May 2001, pp. 253–302.

[35] Hoffmann, J.; Porteous, J.; Sebastia, L. "Ordered Landmarks in Planning", *Journal of Artificial Intelligence Research*, vol. 22–1, Nov 2004, pp. 215–278.

[36] Hong, J. "Goal recognition through goal graph analysis", *Journal of Artificial Intelligence Research*, vol. 15, Jul 2001, pp. 1–30.

[37] Jae Weon Choi; Young Bong See; Wan Suk Yoo; Man Hyung Lee. "LQR approach using Eigenstructure assignment with an active suspension control application". In: Proceedings of the IEEE International Conference on Control Applications, 1998, pp. 1235–1239.

[38] Kambhampati, S. "Model-lite planning for the web age masses: The challenges of planning with incomplete and evolving domain models". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2007, pp. 1601–1604.

[39] Kaminka, G. A.; Vered, M.; Agmon, N. "Plan recognition in continuous domains". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2018, pp. 6202–6210.

[40] Keren, S.; Gal, A.; Karpas, E. "Goal Recognition Design". In: Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS), 2014, pp. 1–8.

[41] Keren, S.; Gal, A.; Karpas, E. "Goal Recognition Design for Non-Optimal Agents". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2015, pp. 3298–3304.

[42] Keren, S.; Gal, A.; Karpas, E. "Goal Recognition Design with Non-Observable Actions". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2016, pp. 3152–3158.

[43] Kerkez, B.; Cox, M. T. "Case-Based Plan Recognition with Incomplete Plan Libraries". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI) Fall Symposium on Intent Inference, 2002, pp. 52–54.

[44] Keyder, E.; Richter, S.; Helmert, M. "Sound and complete landmarks for and/or graphs". In: Proceedings of the European Conference on Artificial Intelligence (ECAI), 2010, pp. 335–340.

[45] Kong, K.; Tomizuka, M. "Nominal model manipulation for enhancement of stability robustness for disturbance observer-based control systems", *International Journal of Control, Automation and Systems*, vol. 11, Jan 2013, pp. 12–20.

[46] Lee, J.-J.; McCartney, R. "Partial Plan Recognition with Incomplete Information". In: Proceedings of International Conference on Multi Agent Systems, 1998, pp. 445–446.

[47] Ljung, L. "System identification", *Signal Analysis and Prediction*, vol. 21, Jan 1998, pp. 163–173.

[48] Masters, P.; Sardiña, S. "Cost-Based Goal Recognition for Path-Planning". In: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2017, pp. 750–758.

[49] Masters, P.; Sardiña, S. "Cost-based goal recognition in navigational domains", *Journal of Artificial Intelligence Research*, vol. 64, Feb 2019, pp. 197–242.

[50] Masters, P.; Sardiña, S. "Goal recognition for rational and irrational agents". In: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS), 2019, pp. 440–448.

[51] McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; Wilkins, D. "PDDL − The Planning Domain Definition Language". In: Proceedings of the International Conference on Artificial Intelligence Planning Systems (AIPS), 1998, pp. 1–8.

[52] Mirsky, R.; Gal, Y. K.; Shieber, S. M. "CRADLE: An Online Plan Recognition Algorithm for Exploratory Domains", *ACM Transactions on Intelligent Systems and Technology*, vol. 8–3, Apr 2017, pp. 45:1–45:22.

[53] Mirsky, R.; Gal, Y. K.; Tolpin, D. "Session analysis using plan recognition". In: The Workshop on User Interfaces and Scheduling and Planning at the International Conference on Automated Planning and Scheduling (ICAPS), 2017, pp. 1–7.

[54] Mirsky, R.; Stern, R.; Gal, K.; Kalech, M. "Sequential plan recognition: An iterative approach to disambiguating between hypotheses", *Artificial Intelligence*, vol. 260, Jul 2018, pp. 51–73.

[55] Mirsky, R.; Stern, R.; Gal, Y. K.; Kalech, M. "Sequential Plan Recognition". In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2016, pp. 401–407.

[56] Mitrovic, D.; Klanke, S.; Vijayakumar, S. "Adaptive optimal feedback control with learned internal dynamics models", *From Motor Learning to Interaction Learning in Robots*, vol. 264, Jan 2010, pp. 65–84.

[57] Montufar, G. F.; Pascanu, R.; Cho, K.; Bengio, Y. "On the number of linear regions of deep neural networks". In: Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS), 2014, pp. 1–9.

[58] M'Sirdi, N.; Rabhi, A.; Naamane, A. "A Nominal Model for Vehicle Dynamics and Estimation of Input Forces and Tire Friction". In: International Conference on Control Systems and Computer Science (CSC), 2007, pp. 1–7.

[59] Nair, V.; Hinton, G. E. "Rectified linear units improve restricted boltzmann machines". In: Proceedings of the International Conference on Machine Learning (ICML), 2010, pp. 807–814.

[60] Nguyen, T.; Sreedharan, S.; Kambhampati, S. "Robust Planning with Incomplete Domain Models", *Artificial Intelligence*, vol. 245, Apr 2017, pp. 134 – 161.

[61] Nguyen, T. A.; Kambhampati, S. "A Heuristic Approach to Planning with Incomplete STRIPS Action Models". In: Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS), 2014, pp. 190–198.

[62] Patra, J. C.; Pal, R. N.; Chatterji, B. N.; Panda, G. "Identification of nonlinear dynamic systems using functional link artificial neural networks", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 29–2, Apr 1999, pp. 254–262.

[63] Pattison, D.; Long, D. "Domain Independent Goal Recognition." In: Proceedings of the Starting AI Researcher Symposium (STAIRS), 2010, pp. 1–10.

[64] Pearl, J. "Causality: Models, Reasoning and Inference". Cambridge University Press, 2009, 1st ed., 464p.

[65] Pereira, R. F.; Meneguzzi, F. "Landmark-Based Plan Recognition". In: Proceedings of the European Conference on Artificial Intelligence (ECAI), 2016, pp. 1706–1707.

[66] Pereira, R. F.; Meneguzzi, F. "Goal and Plan Recognition Datasets using Classical Planning Domains". (Accessed July 2019), https://doi.org/10.5281/zenodo.825878, 2017.

[67] Pereira, R. F.; Meneguzzi, F. "Goal Recognition in Incomplete Domain Models". In: Proceedings of Association for the Advancement of Artificial Intelligence (AAAI), 2018, pp. 8127–8128.

[68] Pereira, R. F.; Meneguzzi, F. "Goal Recognition in Incomplete STRIPS Domain Models". In: The Workshop on Plan, Activity, and Intent Recognition (PAIR) at the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2018, pp. 1–8.

[69] Pereira, R. F.; Oren, N.; Meneguzzi, F. "Detecting Commitment Abandonment by Monitoring Sub-Optimal Steps During Plan Execution". In: Proceedings of the Conference on Autonomous Agents and MultiAgent Systems (AAMAS), 2017, pp. 1685–1687.

[70] Pereira, R. F.; Oren, N.; Meneguzzi, F. "Landmark-Based Heuristics for Goal Recognition". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2017, pp. 3622–3628.

[71] Pereira, R. F.; Oren, N.; Meneguzzi, F. "Monitoring Plan Optimality using Landmarks and Domain-Independent Heuristics". In: The Workshop on Plan, Activity, and Intent Recognition (PAIR) at the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2017, pp. 1–8.

[72] Pereira, R. F.; Oren, N.; Meneguzzi, F. "Landmark-based approaches for goal recognition as planning", *Artificial Intelligence*, vol. 279, Feb 2020, pp. 1–32.

[73] Pereira, R. F.; Oren, N.; Meneguzzi, F. "Using sub-optimal plan detection to identify commitment abandonment in discrete environments", *ACM Transactions on Intelligent Systems and Technology*, vol. 11, Feb 2020, pp. 1–26.

[74] Pereira, R. F.; Pereira, A. G.; Meneguzzi, F. "Landmark-enhanced heuristics for goal recognition in incomplete domain models". In: Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS), 2019, pp. 329–337.

[75] Pereira, R. F.; Vered, M.; Meneguzzi, F.; Ramírez, M. "Online probabilistic goal recognition over nominal models". In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2019, pp. 5547–5553.

[76] Pynadath, D. V.; Wellman, M. P. "Accounting for Context in Plan Recognition, with Application to Traffic Monitoring", *Computing Research Repository (CoRR)*, vol. abs/1302.4980, Aug 2013, pp. 472–481.

[77] Ramírez, M.; Geffner, H. "Plan Recognition as Planning". In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2009, pp. 1778–1783.

[78] Ramírez, M.; Geffner, H. "Probabilistic Plan Recognition Using Off-the-Shelf Classical Planners". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2010, pp. 1121–1126.

[79] Rasmussen, C. E.; Williams, C. K. I. "Gaussian Processes for Machine Learning". MIT Press, 2006, 1st ed., 245p.

[80] Richter, S.; Helmert, M.; Westphal, M. "Landmarks Revisited". In: Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI), 2008, pp. 975–982.

[81] Richter, S.; Westphal, M. "The LAMA Planner: Guiding Cost-based Anytime Planning with Landmarks", *Journal of Artificial Intelligence Research*, vol. 39–1, Jan 2010, pp. 127–177.

[82] Russell, S.; Norvig, P. "Artificial intelligence: A Modern Approach". Prentice Hall, 2010, 3 ed., 1132p.

[83] Sanner, S. "Relational Dynamic Influence Diagram Language (RDDL): Language Description", Technical Report, Australian National University, 2011, 25p.

[84] Say, B.; Sanner, S. "Planning in factored state and action spaces with learned binarized neural network transition models". In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2018, pp. 4815–4821.

[85] Say, B.; Wu, G.; Zhou, Y. Q.; Sanner, S. "Nonlinear hybrid planning with deep net learned transition models and mixed-integer linear programming". In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2017, pp. 750–756.

[86] Schmidt, C. F.; Sridharan, N. S.; Goodson, J. L. "The Plan Recognition Problem: An Intersection of Psychology and Artificial Intelligence", *Journal of Artificial Intelligence Research*, vol. 11–1-2, May 1978, pp. 45–83.

[87] Sohrabi, S.; Riabov, A. V.; Udrea, O. "Plan Recognition as Planning Revisited". In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2016, pp. 3258–3264.

[88] Sukthankar, G.; Goldman, R. P.; Geib, C.; Pynadath, D. V.; Bui, H. H. "Plan, Activity, and Intent Recognition: Theory and Practice". Elsevier, 2014, 1st ed., 424p.

[89] Sutton, R. S.; Barto, A. G. "Reinforcement Learning: An Introduction". USA: A Bradford Book, 2018, 1st ed., 552p.

[90] Tassa, Y.; Erez, T.; Todorov, E. "Synthesis and stabilization of complex behaviours through online trajectory optimization". In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS), 2012, pp. 4906–4913.

[91] Uzan, O.; Dekel, R.; Seri, O.; Gal, Y. K. "Plan Recognition for Exploratory Learning Environments Using Interleaved Temporal Search", *AI Magazine*, vol. 36–2, Aug 2015, pp. 10–21.

[92] Vered, M.; Kaminka, G. A.; Biham, S. "Online goal recognition through mirroring: Humans and agents". In: Proceedings of the Annual Conference on Advances in Cognitive Systems (ACS), 2016, pp. 1–12.

[93] Vered, M.; Pereira, R. F.; Magnaguagno, M.; Meneguzzi, F.; Kaminka, G. A. "Towards Online Goal Recognition Combining Goal Mirroring and Landmarks". In: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2018, pp. 2112–2114.

[94] Vidal, V.; Geffner, H. "Solving Simple Planning Problems with More Inference and No Search". In: Proceedings of the Conference on Principles and Practice of Constraint Programming (CP), 2005, pp. 682–696.

[95] Warren, C. W. "Global path planning using artificial potential fields". In: Proceedings of the International Conference on Robotics and Automation (ICRA), 1989, pp. 316–321.

[96] Weber, C.; Bryce, D. "Planning and Acting in Incomplete Domains". In: Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS), 2011, pp. 1–8.

[97] Wu, G.; Say, B.; Sanner, S. "Scalable planning with tensorflow for hybrid nonlinear domains". In: Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS), 2017, pp. 6273–6283.

[98] Yamaguchi, A.; Atkeson, C. G. "Neural networks and differential dynamic programming for reinforcement learning problems". In: Proceedings of the International Conference on Robotics and Automation (ICRA), 2016, pp. 5434–5441.

[99] Zhang, T. "Solving large scale linear prediction problems using stochastic gradient descent algorithms". In: Proceedings of the International Conference on Machine Learning (ICML), 2004, pp. 919–926.

[100] Zhu, L.; Givan, R. "Landmark extraction via planning graph propagation". In: Proceedings of the Doctoral Consortium at the International Conference on Automated Planning and Scheduling (ICAPS), 2003, pp. 1–7.

[101] Zhuo, H. H. "Recognizing Multi-Agent Plans When Action Models and Team Plans Are Both Incomplete", *ACM Transactions on Intelligent Systems and Technology*, vol. 10–3, May 2019, pp. 1–24.

[102] Zhuo, H. H.; Li, L. "Multi-agent plan recognition with partial team traces and plan libraries". In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2011, pp. 484–489.

[103] Zhuo, H. H.; Nguyen, T. A.; Kambhampati, S. "Refining incomplete planning domain models through plan traces". In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2013, pp. 2451–2458.

[104] Zhuo, H. H.; Yang, Q.; Kambhampati, S. "Action-Model Based Multi-agent Plan Recognition". In: Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS), pp. 377–385.

APPENDIX A – LANDMARK-BASED HEURISTICS FOR GOAL RECOGNITION

In this Appendix, we develop goal recognition heuristics that rely on planning landmarks over *complete and correct domains models*. We next show how we build such heuristics, presenting examples and theoretical properties. Note that we build the goal recognition heuristics presented in Chapter 3 using the concepts developed in this Appendix.

Computing Achieved Landmarks in Observations

An essential part of our heuristic approaches to goal recognition is the ability to track and compute the evidence of achieved fact landmarks in the observations. To do so, we compute the evidence of achieved fact landmarks in preconditions and effects of observed actions during a plan execution using the COMPUTEACHIEVEDLANDMARKS function shown in Algorithm 3. This algorithm takes as input an initial state $\mathcal{I}$, a set of candidate goals $\mathcal{G}$, a sequence of observed actions Obs, and a map $\mathcal{L}_{\mathcal{G}}$ containing candidate goals and their extracted fact landmarks (provided by the EXTRACTLANDMARKS[1] function which computes fact landmarks given a planning domain). Note that Algorithm 3 can be easily modified to allow it to deal with observations as states, so instead of analyzing preconditions and effects of actions, we compare the observations directly to computed landmarks.

Algorithm 3 iterates over the set of candidate goals $\mathcal{G}$ (Line 3) selecting the fact landmarks $\mathcal{L}_G$ of each goal G in $\mathcal{L}_{\mathcal{G}}$ in Line 4 and computes the fact landmarks that are in the initial state in Line 5. With this information, the algorithm iterates over the observed actions Obs to compute the achieved fact landmarks of G in Lines 7 to 10. For each observed action o in Obs, the algorithm computes all fact landmarks of G that are either in the preconditions or effects of o in Line 8. As we deal with partial observations in a plan execution some executed actions may be missing from the observation sequence, thus whenever we identify a fact landmark, we also infer that its predecessors must have been achieved in Line 9. For example, consider that the set of fact landmarks to achieve a goal from a state is represented by the following ordered facts: (at A) $\prec$ (at B) $\prec$ (at C) $\prec$ (at D), and we observe just one action during a plan

---

[1]This landmark extraction algorithm takes as input a planning domain definition $\Xi = \langle \Sigma, \mathcal{A} \rangle$, an initial state $\mathcal{I}$, and a set of candidate goals $\mathcal{G}$ or a single goal G. In case the input is a set of candidate goals $\mathcal{G}$, this function outputs a map $\mathcal{L}_{\mathcal{G}}$ that associates candidate goals to their respective ordered fact landmarks (i.e., a set of landmarks with an order relation).

execution, and this observed action contains the fact landmark (at C) as an effect. From this observed action, we can infer that the predecessors of (at C) must have been achieved before this observation (i.e., (at A) and (at B)). Therefore, we also include them as achieved landmarks. At the end of each iteration over an observed action o, the algorithm stores the set of achieved landmarks of G in $\mathcal{AL}_G$ in Line 10. Finally, after computing the evidence of achieved landmarks in the observations for a candidate goal G, the algorithm stores the set of achieved landmarks $\mathcal{AL}_G$ of G in Λ_G (Line 11) and returns a map $\Lambda_{\mathcal{G}}$ containing all candidate goals and their respective achieved fact landmarks (Line 12). Example APPENDIX A.1 illustrates the execution of Algorithm 3 to compute achieved landmarks from the observations of our running example.

Algorithm 3 Compute Achieved Landmarks in Observations.

Input: $\mathcal{I}$ *initial state,* $\mathcal{G}$ *set of candidate goals,* Obs *observations, and* $\mathcal{L}_{\mathcal{G}}$ *goals and their extracted landmarks.*

Output: *A map of goals to their achieved landmarks.*

1: **function** COMPUTEACHIEVEDLANDMARKS($\mathcal{I}, \mathcal{G}, Obs, \mathcal{L}_{\mathcal{G}}$)
2:     $\Lambda_{\mathcal{G}} := \langle \rangle$
3:     **for each** goal G in $\mathcal{G}$ **do** $\triangleright$ *Map goals $\mathcal{G}$ to their respective achieved landmarks.*
4:         $\mathcal{L}_G :=$ fact landmarks of G s.t $\langle G, \mathcal{L}_G \rangle$ in $\mathcal{L}_{\mathcal{G}}$
5:         $\mathcal{L}_{\mathcal{I}} :=$ all fact landmarks $L \in \mathcal{I}$
6:         $\mathcal{L} := \emptyset$
7:         **for each** observed action o in Obs **do**
8:             $\mathcal{L} := \{L \in \mathcal{L}_G | L \in pre(o) \cup pre(o)^+ \wedge L \notin \mathcal{L}\}$
9:             $\mathcal{L}_{\prec} :=$ predecessors $L_{\prec}$ of all $L \in \mathcal{L}$, s.t $L_{\prec} \notin \mathcal{L}$
10:        $\mathcal{AL}_G := \mathcal{AL}_G \cup \{\mathcal{L}_{\mathcal{I}} \cup \mathcal{L} \cup \mathcal{L}_{\prec}\}$
11:      $\Lambda_{\mathcal{G}}(G) := \mathcal{AL}_G$ $\triangleright$ *Achieved landmarks of G.*
12:     **return** $\Lambda_{\mathcal{G}}$

Example APPENDIX A.1. *Consider the* BLOCKS-WORLD *example from* Figure 2.3*, and the following observed actions:* (unstack E A) *and* (stack E D)*. From these observed actions, the candidate goal* RED*, and the set of fact landmarks of this candidate goal (Figure AP-PENDIX A.1), our algorithm computes that the following fact landmarks have been achieved:*

- $\mathcal{AL}_{\text{RED}} = \{$[(clear R)], [(on E D)],
  [(clear R) (ontable R) (handempty)],
  [(on E A) (clear E) (handempty)],
  [(clear D) (holding E)],
  [(on D B) (clear D) (handempty)]$\}$

In the preconditions of (unstack E A) *the algorithm computes* [(on E A) (clear E) (handempty)]. *Subsequently, in the preconditions and effects of* (stack E D) *the algorithm computes* [(clear D) (holding E)] *and* [(on E D)], *while it computes the other achieved landmarks for the word* RED *from the initial state.* Figure APPENDIX A.1 *shows the set of achieved landmarks for the word* RED *in gray.*
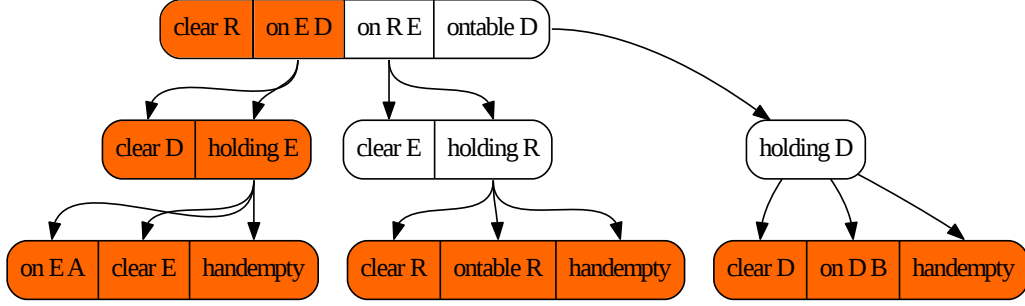
Figure APPENDIX A.1: Ordered fact landmarks extracted for the word RED. Fact landmarks that must be true together are represented by connected boxes. Connected boxes in grey represent achieved fact landmarks. Edges represent prerequisites between landmarks.

The complexity of computing achieved landmarks in observations (Algorithm 3) with the process of extracting landmarks (EL) is: $O(EL + |\mathcal{G}| \cdot |Obs| \cdot |\mathcal{L}_{\mathcal{G}}|)$, where $\mathcal{G}$ is the set of candidate goals, Obs is the observation sequence, and $\mathcal{L}_{\mathcal{G}}$ is the extracted landmarks for $\mathcal{G}$. The complexity of our approach is dominated by the complexity of EL, and thus, given a suitable implementation of EL, our approach has polynomial complexity.

Landmark-Based Goal Completion Heuristic

We now describe a goal recognition heuristic that estimates the percentage of completion of a goal based on the number of landmarks that have been detected, and are required to achieve that goal [70]. This estimate represents the percentage of sub-goals in a goal that have been accomplished based on the evidence of achieved fact landmarks in the observations. We note that a candidate goal is composed of sub-goals comprised of the atomic facts that are part of a conjunction of facts in the goal definition.

Our recognition heuristic estimates the percentage of completion towards a goal by using the set of achieved landmarks computed by Algorithm 3 (COMPUTEACHIEVEDLANDMARKS). Namely, this heuristic operates by aggregating the percentage of completion of each sub-goal into an overall percentage of completion for all facts of a goal. We denote this heuristic as h_{gc},

and it is formally defined by Equation APPENDIX A.1, where $\mathcal{AL}_g$ is the number of achieved landmarks from observations of every sub-goal g of a goal G in $\mathcal{AL}_G$, and $\mathcal{L}_g$ represents the number of necessary landmarks to achieve every sub-goal g of G in $\mathcal{L}_G$.

$$h_{gc}(G, \mathcal{AL}_G, \mathcal{L}_G) = \left(\frac{\sum_{g \in G} \frac{|\mathcal{AL}_g \in \mathcal{AL}_G|}{|\mathcal{L}_g \in \mathcal{L}_G|}}{|G|} \right) \qquad \text{(APPENDIX A.1)}$$

Thus, heuristic h_{gc} estimates the completion of a goal G by calculating the ratio between the sum of the percentage of completion for every sub-goal $g \in G$, i.e., $\sum_{g \in G} \frac{|\mathcal{AL}_g \in \mathcal{AL}_G|}{|\mathcal{L}_g \in \mathcal{L}_G|}$, and the size $|G|$ of the set of sub-goals, that is, the number of sub-goals in G.

Algorithm 4 describes how to recognize goals using the h_{gc} heuristic and takes as input a goal recognition problem T_{GR}, as well as a threshold value θ. The θ threshold gives us flexibility to avoid eliminating candidate goals whose percentage of goal completion are close to the highest completion value. In Line 2, the algorithm uses the EXTRACTLANDMARKS function to extract fact landmarks for all candidate goals. By taking as input the initial state $\mathcal{I}$, the observations O, and the extracted landmarks $\mathcal{L}_\mathcal{G}$, in Line 3, our algorithm first computes the set of achieved landmarks $\Lambda_\mathcal{G}$ for every candidate goal using Algorithm 3. Finally, the algorithm uses the heuristic h_{gc} to estimate goal completion for every candidate G in $\mathcal{G}$, and as output (Line 5), the algorithm returns those candidate goals with the highest estimated value within the threshold θ. Example APPENDIX A.2 shows how heuristic h_{gc} estimates the completion of a candidate goal.

Algorithm 4 Recognize goals using the Goal Completion Heuristic h_{gc}.

Input: Ξ *planning domain definition,* $\mathcal{I}$ *initial state,* $\mathcal{G}$ *set of candidate goals,* Obs *observations, and* θ *threshold.*

Output: *Recognized goal(s).*

1: **function** RECOGNIZE($\Xi, \mathcal{I}, \mathcal{G}, Obs, \theta$)
2: $\quad \mathcal{L}_\mathcal{G} := $ EXTRACTLANDMARKS($\Xi, \mathcal{I}, \mathcal{G}$)
3: $\quad \Lambda_\mathcal{G} := $ COMPUTEACHIEVEDLANDMARKS($\mathcal{I}, \mathcal{G}, Obs, \mathcal{L}_\mathcal{G}$)
4: $\quad maxh := \max\limits_{G' \in \mathcal{G}} h_{gc}(G', \Lambda_\mathcal{G}(G'), \mathcal{L}_\mathcal{G}(G'))$
5: $\quad$ **return** all G s.t $G \in \mathcal{G}$ and
$\qquad\quad h_{gc}(G, \Lambda_\mathcal{G}(G), \mathcal{L}_\mathcal{G}(G)) \geq (maxh - \theta)$

Example APPENDIX A.2. *As an example of how heuristic h_{gc} estimates goal completion of a candidate goal, recall the* BLOCKS-WORLD *example from* Figure 2.3*. Consider that among these candidate goals (*RED*, *BED*, and *SAD*) the correct hidden goal is* RED*, and we observe the following partial sequence of actions: (*unstack E A*) and (*stack E D*). Thus, based on the achieved landmarks $\mathcal{AL}_{RED}$ computed using* Algorithm 3 (Figure APPENDIX A.1)*, our heuristic*

h_{gc} *estimates that the percentage of completion for the goal* RED *is 0.66:* (clear R) $= \frac{1}{1} +$ (on
E D) $= \frac{3}{3} +$ (on R E) $= \frac{1}{3} +$ (ontable D) $= \frac{1}{3}$, *and hence,* $\frac{2.66}{4} = 0.66$. *For the words* BED
and SAD *our heuristic* h_{gc} *estimates respectively, 0.54 and 0.58.*

Besides extracting landmarks for every candidate goal (EL), our landmark-based goal
completion approach iterates over the set of candidate goals $\mathcal{G}$, the observations sequence Obs,
and the extracted landmarks $\mathcal{L}_{\mathcal{G}}$. The heuristic computation of h_{gc} (HC) is linear on the number
of fact landmarks. Thus, the complexity of this approach is: $O(EL + |\mathcal{G}| \cdot |Obs| \cdot |\mathcal{L}_{\mathcal{G}}| + HC)$.
Finally, the goal ranking based on h_{gc} always ensures (under full observability) that the correct
goal ranks highest (i.e., it is sound), with possible ties, as stated in Theorem 1.

Theorem 1 (**Soundness of the h_{gc} Goal Recognition Heuristic**). *Let* $T_{GR} = \langle \Xi, \mathcal{I}, \mathcal{G}, Obs \rangle$
be a goal recognition problem with candidate goals $\mathcal{G}$ *such that* $\forall G_1, G_2 \in \mathcal{G}, G_1 \neq G_2 \rightarrow G_1 \not\subset$
G_2, *a complete and noiseless observation sequence* $Obs = \langle o_1, o_2, ..., o_n \rangle$. *If* $G^* \in \mathcal{G}$ *is the correct
hidden goal, then, for any landmark extraction algorithm that generates fact landmarks* $\mathcal{L}_{\mathcal{G}}$ *and
computed landmarks* $\Lambda_{\mathcal{G}}$, *the estimated value of* h_{gc} *will always be highest for the correct hidden
goal* G^*, *i.e.,* $\forall G \in \mathcal{G}$ *it is the case that* $h_{gc}(G^*, \Lambda_{\mathcal{G}}(G^*), \mathcal{L}_{\mathcal{G}}(G^*)) \geq h_{gc}(G, \Lambda_{\mathcal{G}}(G), \mathcal{L}_{\mathcal{G}}(G))$.

Proof. The proof is straightforward from the definition of fact landmarks ensuring they are
necessary conditions to achieve a goal G and that all facts $g \in G$ are necessary. Let us
first assume that any pair of goals $G_1, G_2 \in \mathcal{G}$ are different, i.e., $G_1 \cap G_2 \neq \emptyset$, and that no
action a in the domain Ξ achieves facts that are in any pair of goals simultaneously. Since
any landmark extraction algorithm includes all facts $g \in G_1$ as landmarks for a goal G_1,
then, for every other goal G_2, there exists at least one fact g such that $g \in G_1 \wedge g \notin G_2$
that sets it apart from G_2. Under these circumstances, an observation sequence Obs for the
correct goal G^* will have achieved a set of landmarks $\Lambda_{\mathcal{G}}(G^*)$ that is exactly the same as the
complete computed set of landmarks $\mathcal{L}_{\mathcal{G}}(G^*)$ for G^*. Hence $h_{gc}(G^*, \Lambda_{\mathcal{G}}(G^*), \mathcal{L}_{\mathcal{G}}(G^*)) = 1$, and
$h_{gc}(G, \Lambda_{\mathcal{G}}(G), \mathcal{L}_{\mathcal{G}}(G)) < 1$ for any other goal $G \in \mathcal{G}$, since the numerator of the h_{gc} computation
will be missing fact g for G as g is not a landmark of G. If we drop the assumption about the
actions not achieving facts simultaneously in any pair of goals or that goals are identical, it is
possible that $h_{gc}(G^*, \Lambda_{\mathcal{G}}(G^*), \mathcal{L}_{\mathcal{G}}(G^*)) = h_{gc}(G, \Lambda_{\mathcal{G}}(G), \mathcal{L}_{\mathcal{G}}(G)) = 1$, which still ensures that the
under h_{gc}, G^* always ranks at the top, possibly tied with other goals. $\square$

Thus, our goal completion heuristic is sound under full observability in the sense that
it can never rank the wrong goal higher than the correct goal when we observe the landmarks.

We note that there is one specific case when our landmark approach can provide wrong rankings, but which we explicitly exclude from the theorem, which is when the set of candidate goals contains two goals such that one is a sub-goal of the other (i.e., $G_1, G_2 \in \mathcal{G}$ and $G_1 \subseteq G_2$). In this case, any kind of "distance" to goal metric will report G_1 as being more likely than G_2 until the observations take the observed agent to G_2 (and these two goals will be tied in the heuristic). We close this section by commenting on the effect of landmark orderings on the accuracy of the heuristic. Specifically, although we do use the landmark order to infer the achievement of necessary prior landmarks that were not observed under missing observations, our heuristic does not consider the actual ordering of the landmarks. We infer prior landmarks to obtain more landmarks when we deal with partial observability. Nevertheless, we have experimented with different scoring mechanisms to account for landmarks having — or not having — been observed in the expected order, and these showed almost no advantage over the current heuristic. Consequently, although there are various different algorithms that generate better landmark orderings [35], the way in which we use the landmarks does not seem to be affected by more or less accurate landmark orderings.

There are two additional properties provable for our h_{gc} heuristic, first, given how our heuristic accounts for landmarks, the value outputted by the heuristic is strictly increasing as observations increase in length.

Proposition 1 (**Monotonicity of h_{gc}**). *The value of h_{gc} is monotonically (non-strictly) increasing in the observation sequence.*

Proof. By definition, $\mathcal{AL}_G$ is monotonically increasing, while all other values in h_{gc} remain constant. Therefore, from Equation APPENDIX A.1, it is clear that h_{gc} must increase. $\qquad\square$

Further, a corollary of Theorem 1 is that, under full observation, only the correct goal can reach a heuristic value of 1. This also illustrates why we restrict the theorem to settings where candidate goals are not subgoals of each other. Consider a goal to be at position d, and another to be at position g, with landmarks a, b, c, d, e, f, g. Since d itself is a landmark of g, d is implicitly a subgoal of g. If we observe all landmarks in an observation, then $h_{gc}(d) = \frac{4}{4} = 1$, and $h_{gc}(g) = \frac{7}{7} = 1$, which leads to Corollary 1.

Corollary 1. *If the goal being recognized has no subgoals being recognized under full observability, then $h_{gc} = 1$ iff the goal the heuristic is recognizing has been achieved.*

Proof. $h_{gc} = 1$ when $\frac{\sum_g \frac{|\mathcal{AL}_G|}{\mathcal{L}}}{|G|} = 1$, which can only occur when $|\mathcal{AL}_g| = |\mathcal{L}_g|$ for all $g \in G$. This clearly occurs when the goal being recognized is achieved. However, if the heuristic is also recognizing a subgoal, then this condition can be satisfied for the subgoal, hence the exception in the proposition. $\square$

Landmark-Based Uniqueness Heuristic

We now turn our attention to another heuristic which uses a measure of the uniqueness of landmarks. Many goal recognition problems contain multiple candidate goals that share common fact landmarks, generating ambiguity for our previous approaches. Clearly, landmarks that are common to multiple candidate goals are less useful for recognizing a goal than landmarks that exist for only a single goal. As a consequence, computing how unique (and thus informative) each landmark is can help disambiguate similar goals for a set of candidate goals. To develop this heuristic based on this intuition, we introduce the concept of *landmark uniqueness*, which is the inverse frequency of a landmark among the landmarks found in a set of candidate goals, and lies in the range (0,1]. For example, consider a landmark L that occurs only for a single goal within a set of candidate goals; since such a landmark is clearly unique, its uniqueness value is maximal (i.e., 1). Equation APPENDIX A.2 formalizes this intuition, describing how the *landmark uniqueness value* is computed for a landmark L and a set of landmarks for goals $\mathcal{L}_\mathcal{G}$.

Using the *landmark uniqueness value*, we estimate which candidate goal is the intended one by summing the uniqueness values of the landmarks achieved in the observations. Unlike our previous heuristic, which estimates progress towards goal completion by analyzing subgoals and their achieved landmarks, the landmark-based uniqueness heuristic estimates the goal completion of a candidate goal G by calculating the ratio between the sum of the uniqueness value of the achieved landmarks of G and the sum of the uniqueness value of all landmarks of G. This algorithm effectively weighs the completion value by the informational value of a landmark so that unique landmarks have the highest weight. To estimate goal completion using the landmark uniqueness value, we calculate the uniqueness value for every extracted landmark in the set of landmarks of the candidate goals using Equation APPENDIX A.2. This computes the landmark uniqueness value of every landmark L of $\mathcal{L}_\mathcal{G}$ and store it into Υ_{uv}. This heuristic is denoted as h_{uniq} and formally defined in Equation APPENDIX A.3.

$$L_{Uniq}(L, \mathcal{L}_{\mathcal{G}}) = \left(\frac{1}{\sum\limits_{\mathcal{L} \in \mathcal{L}_{\mathcal{G}}} |\{L | L \in \mathcal{L}\}|} \right) \qquad \text{(APPENDIX A.2)}$$

$$h_{uniq}(G, \mathcal{AL}_G, \mathcal{L}_G, \Upsilon_{uv}) = \left(\frac{\sum\limits_{\mathcal{A}_L \in \mathcal{AL}_G} \Upsilon_{uv}(\mathcal{A}_L)}{\sum\limits_{L \in \mathcal{L}_G} \Upsilon_{uv}(L)} \right) \qquad \text{(APPENDIX A.3)}$$

Algorithm 5 formalizes a goal recognition function that uses the h_{uniq} heuristic. This algorithm takes as input the same parameters as the previous approach: a goal recognition problem and a threshold θ. Like Algorithm 3, this algorithm extracts the set of landmarks for all candidate goals from the initial state $\mathcal{I}$, stores them in $\mathcal{L}_{\mathcal{G}}$ (Line 2), and computes the set of achieved landmarks based on the observations, storing these in $\Lambda_{\mathcal{G}}$. Unlike Algorithm 4, in Line 6 this algorithm computes the landmark uniqueness value for every landmark L in $\mathcal{L}_{\mathcal{G}}$ and stores it into Υ_{uv}. Finally, using these computed structures, the algorithm recognizes which candidate goal is being pursued from observations using the heuristic h_{uniq}, returning those candidate goals with the highest estimated value within the θ threshold. Example APPENDIX A.3 shows how heuristic h_{uniq} uses the concept of landmark uniqueness value to goal recognition.

Algorithm 5 Recognize goals using Uniqueness Heuristic h_{uniq}.

Input: Ξ *planning domain definition*, $\mathcal{I}$ *initial state*, $\mathcal{G}$ *set of candidate goals*, *Obs observations*, and θ *threshold.*
Output: *Recognized goal(s).*

1: **function** RECOGNIZE($\Xi, \mathcal{I}, \mathcal{G}, Obs, \theta$)
2: $\quad$ $\mathcal{L}_{\mathcal{G}} := $ EXTRACTLANDMARKS($\Xi, \mathcal{I}, \mathcal{G}$)
3: $\quad$ $\Lambda_{\mathcal{G}} := $ COMPUTEACHIEVEDLANDMARKS($\mathcal{I}, \mathcal{G}, Obs, \mathcal{L}_{\mathcal{G}}$)
4: $\quad$ $\Upsilon_{uv} := \langle \rangle$ $\qquad\qquad\qquad$ ▷ *Map of landmarks to their uniqueness value.*
5: $\quad$ **for each** fact landmark L in $\mathcal{L}_{\mathcal{G}}$ **do**
6: $\quad\quad$ $\Upsilon_{uv}(L) := L_{Uniq}(L, \mathcal{L}_{\mathcal{G}})$
7: $\quad$ $maxh := \max\limits_{G' \in \mathcal{G}} h_{uniq}(G', \Lambda_{\mathcal{G}}(G'), \mathcal{L}_{\mathcal{G}}(G'), \Upsilon_{uv})$
8: $\quad$ **return** all G s.t $G \in \mathcal{G}$ and
$\quad\quad\quad h_{uniq}(G, \Lambda_{\mathcal{G}}(G), \mathcal{L}_{\mathcal{G}}(G), \Upsilon_{uv}) \geq (maxh - \theta)$

Example APPENDIX A.3. *Recall the* BLOCKS-WORLD *example from* Figure 2.3 *and consider the following observed actions:* (unstack E A) *and* (stack E D). *Listing APPENDIX A.1 shows the set of extracted fact landmarks for the candidate goals in the* BLOCKS-WORLD *example and their respective uniqueness value. Based on the set of achieved landmarks (shown in bold in* Listing APPENDIX A.1*), our heuristic h_{uniq} estimates the following percentage for each candidate goal: $h_{uniq}(\text{RED}) = \frac{3.66}{6.33} = 0.58$; $h_{uniq}(\text{BED}) = \frac{2.66}{6.33} = 0.42$; and $h_{uniq}(\text{SAD}) = \frac{3.66}{8.33}$*

= 0.44. In this case, Algorithm 5 *correctly estimates* RED *to be the intended goal since it has the highest heuristic value.*

```
- (and (clear B) (on B E) (on E D) (ontable D)) = 6.33
  [(on E D)] = 0.5, [(clear D) (holding E)] = 0.5,
  [(on E A) (clear E) (handempty)] = 0.33, [(ontable D)] = 0.33,
  [(on D B) (clear D) (handempty)] = 0.33, [(holding D)] = 0.33,
  [(clear B) (ontable B) (handempty)] = 1.0, [(on B E)] = 1.0,
  [(clear B)] = 1.0, [(clear E) (holding B)] = 1.0

- (and (clear S) (on S A) (on A D) (ontable D)) = 8.33
  [(clear S)] = 1.0, [(on A D)] = 1.0, [(on S A)] = 1.0,
  [(clear A) (ontable A) (handempty)] = 1.0, [(ontable D)] = 0.33,
  [(clear S) (ontable S) (handempty)] = 1.0, [(holding D)] = 0.33,
  [(on E A) (clear E) (handempty)] = 0.33,
  [(on D B) (clear D) (handempty)] = 0.33,
  [(clear A) (holding S)] = 1.0, [(clear D) (holding A)] = 1.0

- (and (clear R) (on R E) (on E D) (ontable D)) = 6.33
  [(clear R)] = 1.0, [(clear R) (ontable R) (handempty)] = 1.0,
  [(clear D) (holding E)] = 0.5, [(on E D)] = 0.5,
  [(on E A) (clear E) (handempty)] = 0.33, [(ontable D)] = 0.33,
  [(on D B) (clear D) (handempty)] = 0.33, [(holding D)] = 0.33,
  [(on R E)] = 1.0, [(clear E) (holding R)] = 1.0
```

Listing APPENDIX A.1: Extracted fact landmarks for the BLOCKS-WORLD example in Figure 2.3 and their respective uniqueness value.

Similar to our landmark-based goal completion approach, this approach iterates over the set of candidate goals $\mathcal{G}$, the observations sequence Obs, and the extracted landmarks $\mathcal{L}_{\mathcal{G}}$. However, in this approach we weight each landmark by how common this landmark is across all goal hypotheses. We call this weight the uniqueness value $(CLUniq)$ and its computation is linear on the number of landmarks. The heuristic computation of h_{uniq} (HC) is also linear on the number of fact landmarks. Thus, the complexity of this approach is: $O(EL + |\mathcal{G}| \cdot |Obs| \cdot |\mathcal{L}_{\mathcal{G}}| + CLUniq + HC)$. Finally, since this is just a weighted version of the h_{gc} heuristic, it follows trivially from Theorem 1 that, for full observations, h_{uniq} always ranks the correct goal G^* highest.

Corollary 2 (Correctness of h_{uniq} Goal Recognition Heuristic). *Let $T_{GR} = \langle \Xi, \mathcal{I}, \mathcal{G}, Obs \rangle$ be a goal recognition problem with candidate goals $G \in \mathcal{G}$, a complete and noiseless observation sequence $Obs = \langle o_1, o_2, ..., o_n \rangle$. If $G^* \in \mathcal{G}$ is the correct goal, then, for any landmark extraction algorithm that generates fact landmarks $\mathcal{L}_{\mathcal{G}}$ and computed landmarks $\Lambda_{\mathcal{G}}$, the estimated value of h_{uniq} will always be highest for the correct goal G^*, more specifically, $\forall G \in \mathcal{G}$ it is the case that $h_{uniq}(G^*, \Lambda_{\mathcal{G}}(G^*), \mathcal{L}_{\mathcal{G}}(G^*)) \geq h_{uniq}(G, \Lambda_{\mathcal{G}}(G), \mathcal{L}_{\mathcal{G}}(G))$.*

APPENDIX B – GOAL RECOGNITION OVER INCOMPLETE DOMAIN MODELS - DETAILED RESULTS

We now present a detailed experimental evaluation by showing a comparison between our enhanced heuristics (the ones with the best results, i.e., $h_{\widetilde{GC}}$ and $h_{\widetilde{UNIQ}}$ using all types of landmarks, *define*, *possible* and *overlooked* landmarks) and the baseline approaches h_{gc} and h_{uniq} (APPENDIX A) per domain over our recognition datasets with incomplete domain models. We evaluated these approaches over thousands of goal recognition problems using the fifth-ten incomplete domains models, varying the domain incompleteness between 20% and 80%. The results we present here have been used to build the tables and figures in Section 3.4, more specifically, Table 3.2, Figures 3.4a–3.4f (*Correlation*), and Figure 3.5 (*F1-score*).

Tables APPENDIX B.1 and APPENDIX B.2 show the results using the following metrics: recognition time in seconds (*Time*); *Accuracy* (*Acc* %)[1], representing the fraction of time steps in which the correct goal was among the goals found to be most likely, i.e., how good our approaches are for recognizing the correct goal G in $\mathcal{G}$ over time; and *Spread in $\mathcal{G}$* (*S*) represents the average number of returned goals. Each row in the tables express averages for the number of candidate goals $\mathcal{G}$; the percentage of the plan that is actually observed % *Obs*; the average number of observations (actions) per problem *Obs*; and for each approach, the time in seconds to recognize the goal given the observations (*Time*); *Acc* % with which the approaches correctly infer the goal; and S represents the average number of returned goals. Below the name of each domain contains the number of goal recognition problems for all percentage of domain incompleteness.

Tables APPENDIX B.3 and APPENDIX B.4 show a set of inner tables reporting detailed results for all evaluated approaches and domains, varying not only the percentage of domain incompleteness (20%, 40%, 60%, and 80%) but also the percentage of observability (10%, 30%, 50%, 70%, and 100%) of the observation sequence, showing the averages for all types of landmarks, *F1-score*, and *Correlation*.

---

[1]This metric is analogous to the *Quality* metric (also denoted as Q), used for most planning-based goal recognition approaches [77, 78, 19, 87].

#	$\mathcal{G}$	Obs (%)	20% Baseline (h_{gc}) Obs	Time	Acc %	S	$h\widetilde{\ }_{GC}$ (D+P+O) Time	Acc %	S	40% Baseline (h_{gc}) Time	Acc %	S	$h\widetilde{\ }_{GC}$ (D+P+O) Time	Acc %	S	60% Baseline (h_{gc}) Time	Acc %	S	$h\widetilde{\ }_{GC}$ (D+P+O) Time	Acc %	S	80% Baseline (h_{gc}) Time	Acc %	S	$h\widetilde{\ }_{GC}$ (D+P+O) Time	Acc %	S
Blocks (12192) 20.3		10	1.82	0.016	37.7%	1.79	0.117	42.0%	1.45	0.018	40.1%	3.91	0.120	38.6%	1.54	0.009	35.1%	5.36	0.122	44.0%	2.03	0.009	38.3%	5.93	0.124	43.6%	2.15
		30	4.93	0.018	51.2%	1.46	0.138	53.5%	1.16	0.019	45.1%	2.09	0.142	56.4%	1.29	0.008	35.5%	3.78	0.144	61.7%	1.4	0.008	37.8%	4.16	0.145	60.4%	1.35
		50	7.66	0.019	57.7%	1.34	0.159	59.5%	1.11	0.022	53.1%	2.25	0.160	67.8%	1.16	0.008	35.6%	4.07	0.162	70.5%	1.2	0.008	39.7%	4.56	0.163	67.5%	1.2
		70	11.1	0.020	72.6%	1.29	0.176	75.7%	1.07	0.023	62.6%	2.53	0.177	80.4%	1.13	0.009	36.7%	4.36	0.180	84.1%	1.17	0.008	39.2%	4.81	0.181	80.9%	1.14
		100	14.5	0.021	88.4%	1.37	0.181	99.3%	1.12	0.024	71.4%	2.96	0.182	98.6%	1.09	0.007	38.8%	4.75	0.183	99.6%	1.1	0.007	39.9%	5.43	0.184	99.3%	1.12
Campus (900) 2.0		10	1.0	0.001	86.7%	1.04	0.022	100.0%	1.07	0.001	93.3%	1.64	0.022	95.6%	1.13	0.002	100.0%	1.93	0.023	93.3%	1.51	0.001	80.0%	1.51	0.024	95.6%	1.73
		30	2.0	0.001	77.8%	1.07	0.022	100.0%	1.0	0.001	86.7%	1.6	0.023	95.6%	1.09	0.002	95.6%	1.84	0.024	91.1%	1.4	0.001	71.1%	1.4	0.024	95.6%	1.53
		50	3.0	0.001	75.6%	0.98	0.024	100.0%	1.02	0.002	84.4%	1.42	0.023	100.0%	1.04	0.001	93.3%	1.76	0.024	84.4%	1.24	0.002	66.7%	1.33	0.025	91.1%	1.51
		70	4.47	0.002	82.2%	0.98	0.024	100.0%	1.0	0.002	88.9%	1.38	0.025	100.0%	1.0	0.002	95.6%	1.73	0.027	100.0%	1.02	0.002	68.9%	1.33	0.028	95.6%	1.36
		100	5.4	0.002	77.8%	0.98	0.025	100.0%	1.0	0.002	82.2%	1.4	0.025	100.0%	1.0	0.002	93.3%	1.64	0.026	93.3%	1.04	0.001	68.9%	1.33	0.029	95.6%	1.27
Depots (4368) 8.9		10	3.13	0.067	44.0%	1.73	0.122	38.1%	1.48	0.063	49.6%	2.21	0.123	49.6%	2.06	0.063	31.7%	2.25	0.175	74.2%	3.25	0.044	53.2%	3.01	0.135	73.0%	3.37
		30	8.61	0.060	44.4%	1.48	0.143	48.4%	1.14	0.048	56.3%	1.81	0.147	65.5%	1.37	0.040	38.9%	1.97	0.206	78.6%	1.91	0.040	56.3%	2.15	0.171	78.6%	1.85
		50	14.04	0.056	55.6%	1.31	0.274	65.1%	1.1	0.048	65.9%	1.46	0.183	81.3%	1.1	0.042	41.7%	1.65	0.242	83.7%	1.33	0.046	65.1%	1.62	0.250	86.5%	1.34
		70	19.71	0.056	63.1%	1.37	0.312	84.9%	1.03	0.048	71.4%	1.37	0.206	92.1%	1.05	0.042	44.8%	1.69	0.278	91.7%	1.21	0.046	67.1%	1.51	0.242	94.8%	1.21
		100	27.43	0.048	66.7%	1.49	0.345	97.6%	1.06	0.048	77.4%	1.31	0.226	100.0%	1.01	0.036	48.8%	1.79	0.312	95.2%	1.05	0.036	73.8%	1.33	0.274	98.8%	1.04
Driver (4368) 7.1		10	2.61	0.397	45.6%	1.98	1.19	62.3%	1.76	0.321	52.8%	3.71	0.897	56.3%	1.51	0.409	59.9%	3.89	0.812	71.4%	2.52	0.437	48.8%	3.85	0.675	70.2%	2.21
		30	6.96	0.397	44.8%	1.32	1.385	70.2%	1.31	0.306	29.0%	2.01	1.012	68.3%	1.19	0.409	37.7%	2.5	0.913	71.8%	1.51	0.417	31.3%	3.0	0.766	76.6%	1.44
		50	11.18	0.389	56.7%	1.18	1.496	77.4%	1.19	0.302	31.0%	1.71	1.091	79.0%	1.19	0.405	40.9%	2.1	0.988	81.7%	1.36	0.413	21.0%	2.21	0.833	87.3%	1.33
		70	15.64	0.389	63.5%	1.25	1.536	90.1%	1.08	0.302	34.9%	1.6	1.159	90.1%	1.04	0.405	45.6%	1.93	1.044	89.7%	1.13	0.413	20.2%	2.18	0.897	92.5%	1.09
		100	21.71	0.393	69.0%	1.24	1.619	94.0%	1.11	0.298	34.5%	1.7	1.226	94.0%	1.06	0.405	46.4%	1.88	1.107	96.4%	1.14	0.405	13.1%	1.74	0.976	95.2%	1.06
DWR (4368) 7.3		10	5.71	0.175	28.2%	1.48	0.472	38.1%	1.41	0.349	37.7%	2.02	0.716	48.4%	2.29	0.333	32.1%	1.72	0.575	47.6%	2.37	0.290	43.3%	2.48	0.812	64.7%	3.04
		30	16.0	0.163	38.5%	1.27	0.563	63.1%	1.17	0.337	36.9%	1.39	0.893	67.1%	1.5	0.306	43.3%	1.47	0.698	69.8%	1.68	0.274	43.3%	1.73	1.242	75.0%	1.73
		50	26.21	0.163	45.6%	1.25	0.651	72.6%	1.13	0.337	44.4%	1.32	1.041	74.2%	1.3	0.312	50.0%	1.29	0.821	79.4%	1.41	0.274	45.6%	1.46	1.603	82.9%	1.39
		70	36.86	0.163	56.3%	1.29	0.706	89.7%	1.06	0.341	50.4%	1.29	1.135	91.7%	1.16	0.313	62.7%	1.29	0.913	91.3%	1.17	0.278	51.2%	1.27	1.845	92.9%	1.19
		100	51.89	0.167	73.8%	1.4	0.762	100.0%	1.0	0.345	59.5%	1.44	1.256	98.8%	1.01	0.316	79.8%	1.42	1.012	100.0%	1.0	0.278	53.6%	1.2	2.274	100.0%	1.0
Ferry (4368) 7.6		10	2.93	0.016	61.9%	4.4	0.047	77.0%	1.53	0.029	36.5%	2.48	0.048	71.8%	1.38	0.028	73.4%	2.85	0.048	73.4%	1.48	0.020	71.0%	2.86	0.036	81.0%	2.29
		30	7.68	0.012	54.8%	3.99	0.048	91.7%	1.18	0.012	23.4%	1.69	0.052	90.5%	1.17	0.016	82.1%	2.53	0.044	90.1%	1.27	0.012	68.7%	1.64	0.032	90.5%	1.42
		50	12.36	0.008	50.8%	3.91	0.044	98.4%	1.08	0.012	24.2%	1.54	0.052	98.0%	1.07	0.016	86.1%	2.4	0.048	95.6%	1.09	0.016	67.1%	1.38	0.036	93.3%	1.16
		70	17.36	0.008	48.8%	3.93	0.048	100.0%	1.01	0.012	30.2%	1.62	0.056	100.0%	1.01	0.016	90.5%	2.34	0.052	100.0%	1.01	0.012	67.5%	1.28	0.045	98.4%	1.03
		100	24.21	0.012	48.8%	4.15	0.048	100.0%	1.0	0.012	33.3%	1.87	0.06	100.0%	1.0	0.012	90.5%	2.33	0.048	100.0%	1.0	0.012	67.9%	1.49	0.048	100.0%	1.0
Intrusion (5580) 16.7		10	1.92	0.005	18.1%	0.93	0.038	71.7%	1.53	0.006	25.1%	3.88	0.038	58.1%	1.57	0.006	27.6%	3.86	0.025	62.9%	5.46	0.006	27.0%	4.67	0.016	47.9%	6.57
		30	4.48	0.006	22.2%	0.73	0.038	97.5%	1.05	0.006	23.2%	2.84	0.038	79.4%	1.06	0.006	20.6%	2.58	0.025	62.5%	2.41	0.003	28.6%	4.67	0.016	41.6%	4.49
		50	6.7	0.006	23.5%	0.75	0.041	99.4%	1.02	0.006	23.2%	2.56	0.044	90.5%	1.04	0.006	20.3%	1.96	0.029	68.9%	1.81	0.006	28.3%	4.67	0.022	45.7%	3.82
		70	9.55	0.006	24.4%	0.8	0.044	100.0%	1.0	0.006	23.5%	1.86	0.044	93.3%	1.04	0.006	22.5%	1.79	0.032	76.2%	1.82	0.006	28.3%	4.67	0.022	49.2%	2.85
		100	13.07	0.007	27.4%	0.99	0.044	100.0%	1.0	0.006	28.1%	1.36	0.044	94.8%	1.01	0.007	32.6%	2.11	0.037	83.7%	1.79	0.007	28.1%	4.67	0.022	59.3%	2.07
IPC-Grid (8076) 8.7		10	3.18	1.339	26.6%	1.33	0.861	34.1%	1.41	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		30	8.82	1.243	29.0%	1.23	0.897	40.9%	1.05	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		50	14.07	1.191	32.5%	1.22	0.913	42.9%	1.03	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		70	19.86	1.173	35.3%	1.19	0.964	59.1%	1.02	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		100	27.71	1.145	45.2%	1.14	0.993	76.2%	1.01	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
Kitchen (900) 3.0		10	1.33	0.001	55.6%	1.47	0.012	93.3%	2.33	0.001	62.2%	1.64	0.013	82.2%	2.42	0.001	28.9%	0.67	0.014	88.9%	2.67	0.002	42.2%	1.67	0.014	100.0%	2.58
		30	3.33	0.001	46.7%	1.09	0.012	93.3%	1.62	0.001	51.1%	1.42	0.013	86.7%	2.2	0.002	24.4%	0.67	0.014	86.7%	2.64	0.002	51.1%	1.67	0.014	100.0%	2.67
		50	4.0	0.001	40.0%	0.91	0.012	71.1%	1.49	0.001	57.8%	1.38	0.013	73.3%	1.76	0.001	24.4%	0.67	0.013	93.3%	2.64	0.001	51.1%	1.67	0.014	100.0%	2.67
		70	5.0	0.001	44.4%	0.82	0.014	71.1%	1.36	0.002	60.0%	1.38	0.014	71.1%	1.8	0.001	22.2%	0.67	0.015	97.8%	2.62	0.001	55.6%	1.67	0.015	100.0%	2.58
		100	7.47	0.001	46.7%	0.84	0.014	77.8%	1.36	0.001	57.8%	1.33	0.014	73.3%	1.76	0.002	24.4%	0.67	0.015	100.0%	2.58	0.002	51.1%	1.67	0.015	100.0%	2.58
Logistics (8076) 10.5		10	2.82	0.096	18.7%	1.56	0.429	68.6%	3.1	0.115	33.6%	2.63	0.847	82.4%	5.73	0.068	2.0%	0.23	0.804	90.6%	6.81	0.083	2.0%	0.23	1.545	94.3%	7.05
		30	8.01	0.081	21.1%	1.14	0.551	85.2%	2.11	0.105	37.5%	1.79	1.033	90.6%	3.5	0.061	2.0%	0.23	1.059	96.5%	4.19	0.076	2.0%	0.23	1.843	96.3%	3.9
		50	13.07	0.083	21.8%	1.0	0.61	96.7%	1.75	0.107	40.7%	1.55	1.212	93.9%	2.64	0.063	2.0%	0.22	1.261	98.7%	3.07	0.076	2.0%	0.22	2.068	98.3%	2.8
		70	18.33	0.083	22.9%	0.97	0.651	99.3%	1.24	0.107	43.6%	1.5	1.266	99.8%	1.5	0.063	2.0%	0.22	1.344	100.0%	1.61	0.078	2.0%	0.22	2.193	99.8%	1.53
		100	24.41	0.071	23.0%	1.1	0.585	100.0%	1.0	0.093	46.4%	1.62	1.126	100.0%	1.0	0.055	1.6%	0.18	1.219	100.0%	1.0	0.066	1.6%	0.18	1.913	100.0%	1.0
Miconic (4368) 6.0		10	3.96	0.282	56.3%	1.58	0.278	48.0%	1.25	0.361	84.5%	2.21	0.353	57.9%	2.19	0.306	64.7%	2.93	0.294	54.8%	1.84	0.302	68.7%	3.12	0.290	59.1%	2.2
		30	11.14	0.286	81.3%	1.29	0.272	77.4%	1.13	0.339	95.2%	1.67	0.325	77.0%	1.24	0.348	81.7%	1.82	0.336	73.0%	1.17	0.353	83.3%	2.07	0.340	75.0%	1.34
		50	18.07	0.294	94.4%	1.19	0.281	91.3%	1.02	0.341	97.2%	1.31	0.340	88.9%	1.06	0.355	88.9%	1.46	0.348	87.7%	1.05	0.366	87.7%	1.79	0.357	83.7%	1.21
		70	25.32	0.315	98.4%	1.07	0.284	96.8%	1.01	0.347	99.6%	1.13	0.343	97.2%	1.02	0.362	92.5%	1.51	0.349	94.4%	1.04	0.370	94.8%	1.26	0.361	90.9%	1.03
		100	35.57	0.324	100.0%	1.02	0.311	100.0%	1.02	0.351	100.0%	1.02	0.349	100.0%	1.02	0.373	94.0%	1.07	0.355	97.2%	1.02	0.384	100.0%	1.26	0.380	100.0%	1.02
Rovers (4368) 6.0		10	3.0	0.812	47.2%	2.03	1.048	65.1%	2.06	0.944	48.8%	2.23	1.512	64.7%	1.92	1.083	65.5%	3.3	1.401	63.9%	2.4	1.329	50.4%	2.6	1.667	76.6%	3.25
		30	7.93	0.790	59.9%	1.64	1.111	86.1%	1.52	0.933	59.1%	1.69	1.607	78.6%	1.4	1.083	67.9%	2.68	1.542	71.8%	1.64	1.036	55.6%	2.42	1.520	83.7%	2.21
		50	12.75	0.794	70.6%	1.56	1.163	92.5%	1.15	0.929	69.4%	1.5	1.675	92.1%	1.12	1.067	70.6%	2.16	1.675	84.1%	1.21	1.032	56.7%	1.66	1.623	93.7%	1.44
		70	17.96	0.795	84.1%	1.36	1.206	99.6%	1.05	0.933	86.1%	1.4	1.726	99.6%	1.05	1.071	77.0%	2.04	1.782	93.7%	1.06	1.103	61.5%	1.45	1.718	99.2%	1.08
		100	24.93	0.786	90.5%	1.3	1.226	100.0%	1.0	0.929	91.7%	1.36	1.786	100.0%	1.0	1.238	76.2%	1.77	1.869	98.8%	1.01	1.036	66.7%	1.11	1.845	100.0%	1.0
Satellite (4368) 6.4		10	2.07	0.119	71.8%	2.74	0.258	76.6%	2.32	0.127	25.0%	1.26	0.286	83.3%	2.76	0.123	52.4%	2.21	0.214	75.8%	2.33	0.135	56.0%	2.56	0.238	80.6%	2.63
		30	5.43	0.115	74.6%	1.95	0.286	88.5%	1.44	0.119	26.2%	0.74	0.321	88.9%	1.6	0.119	53.2%	1.63	0.238	86.5%	1.44	0.131	56.3%	1.35	0.272	85.7%	1.51
		50	8.71	0.115	78.6%	1.66	0.31	93.3%	1.23	0.119	29.0%	0.63	0.349	92.9%	1.28	0.119	56.3%	1.4	0.258	91.7%	1.27	0.131	58.7%	1.04	0.298	92.1%	1.3
		70	12.29	0.115	86.5%	1.57	0.329	96.0%	1.07	0.119	32.1%	0.6	0.377	96.8%	1.09	0.119	59.5%	1.23	0.278	96.4%	1.08	0.131	63.1%	0.84	0.325	94.8%	1.09
		100	16.89	0.107	89.3%	1.38	0.345	98.8%	1.01	0.119	35.7%	0.6	0.405	100.0%	1.02	0.119	63.1%	1.15	0.298	100.0%	1.02	0.131	67.9%	0.8	0.357	100.0%	1.04
Sokoban (4368) 7.1		10	3.18	1.339	26.6%	1.33	0.861	34.1%	1.41	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		30	8.82	1.243	29.0%	1.23	0.897	40.9%	1.05	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		50	14.07	1.191	32.5%	1.22	0.913	42.9%	1.03	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		70	19.86	1.173	35.3%	1.19	0.964	59.1%	1.02	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		100	27.71	1.145	45.2%	1.14	0.993	76.2%	1.01	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
Zeno (4368) 6.9		10	2.61	1.183	44.4%	1.59	1.651	69.0%	2.06	1.560	41.7%	2.62	2.071	59.9%	1.9	1.567	57.9%	3.34	2.179	73.0%	2.26	1.329	69.8%	4.41	1.667	79.4%	2.67
		30	6.75	1.163	44.4%	1.38	1.718	78.6%	1.54	1.468	18.3%	1.71	1.762	66.3%	1.31	1.540	43.3%	2.57	1.817	80.2%	1.64	1.278	52.8%	3.38	1.895	88.5%	1.78
		50	10.82	1.151	56.7%	1.33	1.786	88.1%	1.13	1.476	15.9%	1.76	1.889	82.9%	1.13	1.532	46.4%	2.63	1.933	90.5%	1.13	1.286	48.0%	3.27	2.254	94.0%	1.2
		70	15.21	1.155	59.9%	1.45	1.845	96.0%	1.06	1.476	10.7%	2.18	1.944	92.5%	1.02	1.532	39.3%	2.6	1.938	96.4%	1.01	1.290	42.9%	3.19	2.233	97.2%	1.02
		100	21.14	1.179	67.9%	1.83	1.905	98.8%	1.0	1.488	10.7%	3.2	2.183	98.8%	1.0	1.524	39.3%	3.02	2.247	98.8%	1.0	1.286	39.3%	3.42	2.314	100.0%	1.0

Table APPENDIX B.1: Experimental results of our enhanced goal completion heuristic (using *definite D*, *possible P*, and *overlooked O* landmarks) against the baseline (goal completion) for recognizing goals in incomplete STRIPS domain models.

Table header: **Incompleteness of $\widetilde{\mathcal{D}}$ (%)** — columns grouped by 20%, 40%, 60%, 80%, each split into **Baseline (h_{uniq})** and **$h_{\widetilde{UNIQ}}$ (D+P+O)**, each reporting Time, Acc %, S.

#	G	Obs (%)	Obs	20% Base Time	20% Base Acc %	20% Base S	20% UNIQ Time	20% UNIQ Acc %	20% UNIQ S	40% Base Time	40% Base Acc %	40% Base S	40% UNIQ Time	40% UNIQ Acc %	40% UNIQ S	60% Base Time	60% Base Acc %	60% Base S	60% UNIQ Time	60% UNIQ Acc %	60% UNIQ S	80% Base Time	80% Base Acc %	80% Base S	80% UNIQ Time	80% UNIQ Acc %	80% UNIQ S
Blocks (12192)	20.3	10	1.82	0.019	30.6%	1.21	0.067	22.2%	1.72	0.012	35.1%	3.93	0.069	30.8%	2.96	0.011	35.0%	6.6	0.073	36.0%	2.25	0.009	39.0%	7.3	0.086	35.9%	3.9
		30	4.93	0.018	51.2%	1.17	0.142	43.5%	1.05	0.011	39.7%	1.98	0.076	43.0%	1.07	0.009	31.3%	4.43	0.075	46.1%	1.06	0.008	30.9%	4.27	0.066	51.6%	1.05
		50	7.66	0.018	58.9%	1.19	0.164	53.4%	1.04	0.011	44.6%	1.96	0.089	54.7%	1.02	0.009	30.4%	4.3	0.092	55.0%	1.02	0.009	28.0%	3.96	0.081	59.5%	1.04
		70	11.1	0.019	74.0%	1.2	0.183	70.2%	1.04	0.012	56.0%	2.38	0.103	67.5%	1.03	0.009	32.0%	4.61	0.107	70.2%	1.02	0.009	29.8%	4.47	0.095	73.8%	1.04
		100	14.5	0.018	90.9%	1.34	0.185	85.1%	1.11	0.011	68.1%	2.7	0.109	84.8%	1.08	0.011	33.3%	4.72	0.116	87.7%	1.06	0.007	31.5%	4.84	0.101	93.5%	1.11
Campus (900)	2.0	10	1.0	0.001	95.6%	1.27	0.022	100.0%	1.07	0.001	93.3%	1.64	0.023	100.0%	1.13	0.002	100.0%	1.93	0.023	86.7%	1.53	0.002	95.6%	1.8	0.025	82.2%	1.58
		30	2.0	0.001	86.7%	1.27	0.022	100.0%	1.0	0.002	86.7%	1.6	0.026	95.6%	1.09	0.001	95.6%	1.84	0.027	88.9%	1.42	0.001	86.7%	1.69	0.028	84.4%	1.42
		50	3.0	0.001	86.7%	1.11	0.022	100.0%	1.02	0.001	86.7%	1.4	0.023	100.0%	1.04	0.002	93.3%	1.76	0.024	73.3%	1.27	0.002	77.8%	1.64	0.025	75.6%	1.31
		70	4.47	0.001	91.1%	1.16	0.024	100.0%	1.0	0.002	88.9%	1.36	0.023	100.0%	1.0	0.002	95.6%	1.73	0.023	73.3%	1.02	0.002	82.2%	1.62	0.024	68.9%	1.22
		100	5.4	0.001	86.7%	1.16	0.025	100.0%	1.0	0.002	88.9%	1.4	0.026	100.0%	1.0	0.003	93.3%	1.64	0.027	73.3%	1.0	0.002	84.4%	1.62	0.028	66.7%	1.18
Depots (4368)	8.9	10	3.13	0.067	58.7%	3.43	0.191	45.2%	1.97	0.063	67.5%	4.06	0.135	46.0%	2.22	0.052	73.8%	5.92	0.175	64.7%	3.64	0.044	69.4%	4.71	0.143	65.5%	3.78
		30	8.61	0.056	60.3%	2.9	0.234	50.8%	1.05	0.048	72.2%	3.46	0.159	57.9%	1.18	0.048	79.8%	5.35	0.214	67.1%	1.81	0.049	72.2%	3.79	0.179	68.3%	1.86
		50	14.04	0.056	71.8%	2.47	0.274	71.4%	1.06	0.048	85.3%	3.23	0.199	77.8%	1.02	0.042	77.4%	4.67	0.252	80.2%	1.29	0.042	84.9%	3.32	0.218	85.7%	1.36
		70	19.71	0.056	74.6%	2.25	0.306	83.7%	1.03	0.052	90.9%	3.23	0.218	86.9%	1.01	0.041	77.8%	4.45	0.282	85.3%	1.14	0.048	89.3%	3.29	0.254	93.7%	1.18
		100	27.43	0.048	77.4%	2.24	0.333	95.2%	1.06	0.048	97.6%	3.15	0.238	96.4%	1.01	0.036	79.8%	4.37	0.319	94.0%	1.07	0.036	96.4%	3.17	0.286	100.0%	1.05
Driver (4368)	7.1	10	2.61	0.397	49.2%	2.02	1.194	56.0%	1.93	0.311	53.2%	3.68	0.921	54.4%	1.96	0.548	68.7%	4.39	0.901	59.5%	2.33	0.429	72.6%	5.48	0.683	55.6%	2.23
		30	6.96	0.393	39.3%	1.19	1.389	46.8%	1.1	0.306	28.2%	1.84	1.032	44.8%	1.11	0.536	44.8%	2.98	1.098	43.7%	1.21	0.413	52.8%	4.44	0.766	50.8%	1.28
		50	11.18	0.389	51.6%	1.11	1.488	59.1%	1.05	0.298	28.6%	1.52	1.111	61.9%	1.08	0.536	48.8%	2.58	1.063	63.5%	1.21	0.409	34.1%	3.17	0.837	68.3%	1.25
		70	15.64	0.389	62.7%	1.21	1.552	81.0%	1.08	0.298	30.6%	1.4	1.187	81.3%	1.08	0.536	80.2%	2.44	1.123	80.2%	1.12	0.409	26.6%	2.73	0.901	86.5%	1.1
		100	21.71	0.381	71.4%	1.27	1.631	89.3%	1.12	0.298	31.0%	1.57	1.214	88.1%	1.1	0.536	52.4%	2.4	1.179	90.5%	1.15	0.405	13.1%	1.9	0.964	91.7%	1.08
DWR (4368)	7.3	10	5.71	0.171	32.9%	1.91	0.512	32.9%	1.13	0.341	66.7%	4.16	0.714	49.2%	2.35	0.325	44.0%	2.54	0.579	36.9%	1.72	0.278	83.3%	5.77	0.825	56.3%	3.15
		30	16.0	0.163	45.2%	1.64	0.603	59.1%	1.05	0.337	61.9%	3.02	0.885	57.1%	1.32	0.317	52.8%	2.07	0.714	52.8%	1.2	0.266	79.4%	4.49	1.254	59.9%	1.46
		50	26.21	0.159	52.8%	1.63	0.691	67.5%	1.04	0.333	64.7%	2.75	1.036	65.5%	1.13	0.319	57.5%	2.05	0.821	63.9%	1.1	0.266	77.8%	3.89	1.603	62.7%	1.15
		70	36.86	0.163	59.1%	1.58	0.74	86.5%	1.02	0.333	64.7%	2.63	1.143	81.7%	1.05	0.313	71.8%	2.12	0.913	79.4%	1.04	0.266	82.5%	3.51	1.865	80.6%	1.02
		100	51.89	0.167	77.4%	1.67	0.798	98.8%	1.0	0.345	75.0%	2.74	1.262	95.2%	1.01	0.321	91.7%	2.36	1.024	96.4%	1.0	0.274	82.1%	3.21	2.202	96.4%	1.0
Ferry (4368)	7.6	10	2.93	0.016	89.7%	6.74	0.04	59.9%	1.35	0.021	44.8%	3.1	0.056	59.1%	1.21	0.028	75.8%	3.61	0.048	61.9%	1.56	0.025	71.4%	3.31	0.036	64.3%	1.94
		30	7.68	0.012	80.6%	6.15	0.042	78.6%	1.06	0.016	26.2%	2.03	0.052	71.0%	1.05	0.016	81.0%	3.2	0.044	77.0%	1.09	0.016	69.0%	2.0	0.032	73.8%	1.15
		50	12.36	0.012	75.0%	5.86	0.044	83.3%	1.02	0.016	29.0%	1.72	0.052	73.8%	1.05	0.016	83.3%	3.17	0.048	79.8%	1.04	0.016	63.9%	1.63	0.036	76.2%	1.07
		70	17.36	0.012	73.4%	5.85	0.048	93.7%	1.02	0.016	31.0%	1.82	0.056	80.6%	1.02	0.016	94.0%	3.15	0.052	93.3%	1.02	0.016	71.8%	1.57	0.041	90.9%	1.02
		100	24.21	0.012	72.6%	6.04	0.048	100.0%	1.0	0.012	39.3%	2.08	0.06	81.0%	1.01	0.012	100.0%	3.14	0.06	100.0%	1.0	0.012	72.6%	1.76	0.048	97.6%	1.02
Intrusion (5580)	16.7	10	1.92	0.005	64.8%	9.09	0.041	54.6%	1.23	0.006	97.1%	15.71	0.038	48.9%	1.32	0.006	92.7%	14.87	0.025	52.4%	5.17	0.006	100.0%	16.67	0.016	43.2%	6.56
		30	4.48	0.006	63.5%	7.55	0.041	87.9%	1.04	0.006	94.3%	14.44	0.038	73.7%	1.03	0.006	86.3%	13.32	0.025	56.2%	2.1	0.006	100.0%	16.67	0.016	32.4%	4.47
		50	6.7	0.006	59.7%	6.76	0.044	95.6%	1.01	0.006	94.0%	14.08	0.044	83.5%	1.02	0.006	85.1%	12.54	0.032	62.9%	1.59	0.006	100.0%	16.67	0.019	29.2%	3.7
		70	9.55	0.006	58.7%	6.43	0.048	97.1%	1.01	0.006	95.2%	13.23	0.044	87.3%	1.02	0.006	87.3%	12.3	0.032	67.9%	1.57	0.006	100.0%	16.67	0.022	27.0%	2.85
		100	13.07	0.007	60.7%	6.52	0.052	100.0%	1.0	0.007	100.0%	12.61	0.052	93.3%	1.03	0.007	97.0%	12.61	0.037	74.8%	1.58	0.007	100.0%	16.67	0.022	29.6%	2.07
IPC-Grid (8076)	8.7	10	3.18	1.339	26.6%	1.33	0.861	34.1%	1.41	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		30	8.82	1.243	29.0%	1.23	0.897	40.9%	1.05	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		50	14.07	1.191	32.5%	1.22	0.913	42.9%	1.03	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		70	19.86	1.173	35.3%	1.19	0.964	59.1%	1.02	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		100	27.71	1.145	45.2%	1.14	0.993	76.2%	1.01	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
Kitchen (900)	3.0	10	1.33	0.001	100.0%	2.69	0.010	93.3%	2.42	0.001	88.9%	2.64	0.011	82.2%	2.42	0.001	77.8%	2.33	0.011	88.9%	2.67	0.002	100.0%	3.0	0.011	100.0%	2.67
		30	3.33	0.001	95.6%	2.24	0.010	95.6%	1.71	0.002	91.1%	2.42	0.010	86.7%	2.2	0.002	82.2%	2.33	0.012	84.4%	2.67	0.001	100.0%	3.0	0.012	100.0%	2.67
		50	4.0	0.001	82.2%	2.02	0.012	73.3%	1.64	0.001	84.4%	2.38	0.012	73.3%	1.76	0.001	75.6%	2.33	0.013	91.1%	2.67	0.002	100.0%	3.0	0.013	100.0%	2.58
		70	5.0	0.001	84.4%	1.89	0.012	73.3%	1.53	0.001	84.4%	2.38	0.012	71.1%	1.8	0.001	73.3%	2.33	0.013	93.3%	2.67	0.001	100.0%	3.0	0.012	100.0%	2.58
		100	7.47	0.001	88.9%	1.93	0.012	77.8%	1.51	0.001	84.4%	2.33	0.012	73.3%	1.76	0.001	75.6%	2.33	0.013	91.1%	2.67	0.001	100.0%	3.0	0.013	100.0%	2.58
Logistics (8076)	10.5	10	2.6	0.087	91.5%	8.89	0.436	59.0%	3.59	0.111	75.6%	6.8	0.876	82.1%	6.31	0.063	99.3%	10.4	0.815	83.4%	6.99	0.078	99.3%	10.4	1.481	92.4%	7.17
		30	7.48	0.078	86.9%	7.84	0.556	68.8%	1.38	0.102	70.2%	4.98	1.078	78.4%	2.97	0.061	99.6%	10.42	1.074	79.3%	3.6	0.076	99.6%	10.42	1.793	87.6%	3.78
		50	12.16	0.078	85.4%	7.46	0.619	83.7%	1.11	0.102	68.2%	4.47	1.194	80.7%	2.03	0.061	98.5%	10.3	1.248	82.4%	2.58	0.074	98.5%	10.3	2.004	88.9%	2.66
		70	17.02	0.081	85.8%	7.46	0.662	89.5%	1.04	0.111	73.2%	4.53	1.251	86.3%	1.2	0.063	98.0%	10.25	1.318	85.6%	1.31	0.074	98.0%	10.25	2.083	87.1%	1.34
		100	22.7	0.066	87.4%	7.41	0.59	100.0%	1.0	0.087	78.1%	4.51	1.104	100.0%	1.0	0.055	100.0%	10.21	1.158	100.0%	1.0	0.066	98.4%	10.21	1.885	100.0%	1.0
Miconic (4368)	6.0	10	3.96	0.280	81.0%	4.08	0.587	53.6%	1.28	0.524	90.9%	4.8	0.738	65.9%	2.23	0.444	99.6%	5.92	0.631	59.5%	1.84	0.317	88.1%	4.47	0.542	60.3%	2.2
		30	11.14	0.293	86.5%	3.89	0.631	77.4%	1.12	0.516	94.0%	4.41	0.791	75.4%	1.25	0.444	99.2%	5.82	0.687	73.8%	1.17	0.312	95.2%	4.38	0.595	75.8%	1.33
		50	18.07	0.286	90.5%	3.69	0.655	92.9%	1.03	0.516	96.8%	4.35	0.825	84.5%	1.06	0.433	98.4%	5.79	0.726	87.3%	1.05	0.313	98.0%	4.36	0.639	85.3%	1.2
		70	25.32	0.286	91.7%	3.66	0.683	97.2%	1.01	0.528	99.6%	4.34	0.853	90.9%	1.02	0.437	98.4%	5.78	0.762	94.4%	1.04	0.306	99.6%	4.33	0.675	94.4%	1.02
		100	35.57	0.286	94.0%	3.67	0.702	100.0%	1.0	0.512	100.0%	4.33	0.869	92.9%	1.0	0.429	97.6%	5.76	0.786	97.6%	1.0	0.319	100.0%	4.33	0.714	100.0%	1.0
Rovers (4368)	6.0	10	3.0	0.941	38.1%	1.65	1.202	48.4%	1.52	1.087	54.8%	2.07	1.532	52.8%	1.62	1.075	72.2%	3.7	1.562	60.7%	2.24	1.306	68.7%	4.16	1.679	62.7%	2.86
		30	7.93	0.929	52.8%	1.4	1.266	66.7%	1.15	1.075	58.7%	1.45	1.627	63.1%	1.09	1.052	68.7%	2.93	1.698	64.7%	1.48	1.020	67.5%	3.63	1.527	65.1%	1.76
		50	12.75	0.949	70.6%	1.32	1.313	80.2%	1.02	1.067	71.0%	1.22	1.706	84.5%	1.03	1.056	73.4%	2.46	1.829	76.6%	1.04	1.061	73.8%	3.04	1.631	75.4%	1.04
		70	17.96	0.929	85.7%	1.31	1.365	92.9%	1.0	1.067	85.7%	1.26	1.766	95.2%	1.01	1.068	78.2%	2.41	1.942	90.1%	1.02	1.021	80.6%	2.95	1.726	91.7%	1.02
		100	24.93	0.929	94.0%	1.3	1.393	100.0%	1.0	1.078	88.1%	1.26	1.821	100.0%	1.0	1.066	85.7%	2.31	2.024	100.0%	1.0	1.012	95.2%	2.89	1.869	98.8%	1.0
Satellite (4368)	6.4	10	2.07	0.111	69.8%	2.72	0.258	61.1%	2.29	0.123	87.7%	5.14	0.286	65.5%	2.49	0.139	82.5%	4.35	0.210	58.7%	2.26	0.135	80.6%	4.35	0.234	65.1%	2.5
		30	5.43	0.107	75.0%	2.02	0.286	71.4%	1.27	0.119	86.5%	4.6	0.321	70.2%	1.39	0.135	84.5%	3.73	0.234	70.2%	1.28	0.135	84.1%	3.08	0.266	73.4%	1.32
		50	8.71	0.103	83.3%	1.84	0.306	77.0%	1.08	0.119	90.9%	4.52	0.349	69.8%	1.12	0.135	87.3%	3.53	0.258	79.4%	1.1	0.131	85.3%	2.81	0.294	79.8%	1.09
		70	12.29	0.107	90.5%	1.81	0.325	87.3%	1.04	0.119	94.0%	4.51	0.377	82.5%	1.07	0.135	88.5%	3.33	0.274	90.5%	1.05	0.131	89.7%	2.64	0.321	91.3%	1.05
		100	16.89	0.107	98.8%	1.67	0.345	89.3%	1.0	0.119	97.6%	4.44	0.405	85.7%	1.02	0.135	96.4%	3.3	0.298	96.4%	1.02	0.131	96.4%	2.65	0.345	100.0%	1.04
Sokoban (4368)	7.1	10	3.18	1.339	26.6%	1.33	0.861	34.1%	1.41	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		30	8.82	1.243	29.0%	1.23	0.897	40.9%	1.05	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		50	14.07	1.191	32.5%	1.22	0.913	42.9%	1.03	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		70	19.86	1.173	35.3%	1.19	0.964	59.1%	1.02	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
		100	27.71	1.145	45.2%	1.14	0.993	76.2%	1.01	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-	Timeout	-	-
Zeno (4368)	6.9	10	2.61	1.167	40.5%	1.64	1.647	44.0%	1.49	1.548	43.3%	3.0	2.059	46.0%	1.79	1.548	61.1%	3.92	2.006	54.0%	2.03	1.298	77.8%	5.29	1.926	67.1%	2.52
		30	6.75	1.131	44.0%	1.3	1.706	54.0%	1.12	1.492	18.3%	1.82	2.004	51.2%	1.06	1.504	47.6%	3.01	2.099	59.9%	1.3	1.294	58.3%	4.08	2.079	68.7%	1.39
		50	10.82	1.127	56.0%	1.29	1.878	68.7%	1.03	1.492	15.5%	1.76	2.167	68.7%	1.02	1.513	48.8%	3.04	2.156	71.4%	1.06	1.282	53.6%	3.88	2.108	79.0%	1.01
		70	15.21	1.135	64.7%	1.42	1.837	77.0%	1.03	1.476	11.5%	2.08	2.292	73.8%	1.04	1.516	42.9%	3.02	2.248	79.4%	1.06	1.286	48.4%	3.8	2.306	88.9%	1.02
		100	21.14	1.143	67.9%	1.8	2.181	86.9%	1.01	1.464	10.7%	3.0	2.395	92.9%	1.0	1.556	42.9%	3.45	2.343	88.1%	1.04	1.286	44.0%	4.02	2.424	96.4%	1.0

Table APPENDIX B.2: Experimental results of our enhanced uniqueness heuristic (using *definite D*, *possible P*, and *overlooked O* landmarks) against the baseline (uniqueness) for recognizing goals in incomplete STRIPS domain models.

10% of Observability

	Domain Incompleteness 20%						Domain Incompleteness 40%						Domain Incompleteness 60%						Domain Incompleteness 80%					
	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO
Baseline (h_{gc})	11.5	0.0	0.0	1.77	0.33	-0.04/0/0	4.9	0.0	0.0	2.41	0.28	-0.02/0/0	3.5	0.0	0.0	2.45	0.25	0.34/0/0	2.9	0.0	0.0	2.79	0.26	0.36/0/0
Baseline (h_{uniq})	11.5	0.0	0.0	1.36	0.33	-0.11/0/0	4.9	0.0	0.0	1.67	0.29	-0.02/0/0	3.5	0.0	0.0	1.42	0.28	0.08/0/0	2.9	0.0	0.0	1.90	0.28	0.09/0/0
$\widetilde{GC}$ (D+P+O)	11.6	1.4	1.3	1.94	0.46	-0.11/-0.08/-0.61	9.6	2.1	1.2	2.24	0.44	0.16/-0.13/-0.47	7.1	2.2	1.4	2.86	0.40	0.31/0.16/-0.26	6.8	1.1	1.3	3.27	0.39	0.21/-0.40/-0.24
$\widetilde{GC}$ (D+O)	11.6	0.0	1.7	2.03	0.45	-0.13/0/-0.55	9.6	0.0	1.7	2.51	0.44	0.15/0/-0.42	7.1	0.0	1.9	3.23	0.40	0.28/0/-0.16	6.8	0.0	1.6	3.58	0.38	0.20/0/-0.23
$\widetilde{GC}$ (P+O)	0.0	1.4	3.0	4.30	0.30	0/-0.16/-0.37	0.0	2.1	2.4	4.11	0.31	0/-0.14/-0.26	0.0	2.2	2.0	4.08	0.29	0/0.01/-0.27	0.0	1.1	1.8	4.26	0.28	0/-0.55/-0.18
$\widetilde{GC}$ (P)	0.0	1.4	0.0	2.31	0.24	0.11/0/0	0.0	2.1	0.0	2.29	0.22	-0.07/0/0	0.0	2.2	0.0	2.08	0.22	0.28/0/0	0.0	1.1	0.0	2.20	0.21	-0.29/0/0
$\widetilde{GC}$ (O)	0.0	0.0	3.4	3.54	0.43	0/0/-0.39	0.0	0.0	2.9	3.92	0.42	0/0/-0.34	0.0	0.0	2.6	4.38	0.38	0/0/-0.16	0.0	0.0	2.1	4.51	0.37	0/0/-0.19
$\widetilde{UNIQ}$ (D+P+O)	11.6	1.4	1.3	1.87	0.40	-0.12/-0.02/-0.46	9.6	2.1	1.2	2.01	0.40	0.08/-0.16/-0.39	7.1	2.2	1.4	2.34	0.35	0.24/0.12/-0.23	6.9	1.1	1.3	2.91	0.35	0.13/-0.57/-0.26
$\widetilde{UNIQ}$ (D+O)	11.6	0.0	1.7	2.00	0.38	-0.13/0/-0.43	9.6	0.0	1.7	2.39	0.38	0.06/0/-0.39	7.1	0.0	1.9	3.01	0.35	0.16/0/-0.26	6.9	0.0	1.6	3.12	0.34	0.12/0/-0.31
$\widetilde{UNIQ}$ (P+O)	0.0	1.4	3.0	4.38	0.30	0/-0.23/-0.47	0.0	2.1	2.4	3.98	0.32	0/-0.33/-0.46	0.0	2.2	2.0	3.79	0.29	0/-0.15/-0.34	0.0	1.1	1.8	3.73	0.29	0/-0.62/-0.35
$\widetilde{UNIQ}$ (P)	0.0	1.4	0.0	1.25	0.26	-0.26/0/0	0.0	2.1	0.0	1.32	0.27	-0.42/0/0	0.0	2.2	0.0	1.32	0.27	-0.20/0/0	0.0	1.1	0.0	1.32	0.27	-0.55/0/0
$\widetilde{UNIQ}$ (O)	0.0	0.0	3.4	3.37	0.43	0/0/-0.42	0.0	0.0	2.9	3.80	0.42	0/0/-0.34	0.0	0.0	2.6	4.24	0.38	0/0/-0.17	0.0	0.0	2.1	4.40	0.37	0/0/-0.19

30% of Observability

	Domain Incompleteness 20%						Domain Incompleteness 40%						Domain Incompleteness 60%						Domain Incompleteness 80%					
	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO
Baseline (h_{gc})	11.5	0.0	0.0	1.46	0.33	0.14/0/0	4.8	0.0	0.0	1.64	0.28	0.29/0/0	3.5	0.0	0.0	1.90	0.25	0.55/0/0	2.9	0.0	0.0	2.16	0.26	0.59/0/0
Baseline (h_{uniq})	11.5	0.0	0.0	1.33	0.33	0.15/0/0	4.8	0.0	0.0	1.59	0.29	0.20/0/0	3.5	0.0	0.0	1.72	0.28	0.21/0/0	2.9	0.0	0.0	2.07	0.28	0.30/0/0
$\widetilde{GC}$ (D+P+O)	11.5	1.4	4.4	1.39	0.66	0.09/0.05/-0.64	9.6	2.1	4.0	1.55	0.63	0.48/0.16/-0.45	7.1	2.2	4.4	1.88	0.57	0.70/0.54/-0.04	6.8	1.1	4.2	2.15	0.55	0.53/-0.08/0.10
$\widetilde{GC}$ (D+O)	11.6	0.0	5.5	1.41	0.65	0.09/0/-0.56	9.6	0.0	5.4	1.64	0.63	0.50/0/-0.35	7.1	0.0	6.0	2.01	0.57	0.69/0/0.16	6.8	0.0	5.1	2.29	0.55	0.54/0/0.15
$\widetilde{GC}$ (P+O)	0.0	1.4	9.5	4.88	0.32	0/0/-0.41	0.0	2.1	7.6	4.49	0.34	0/0/-0.29	0.0	2.2	6.5	4.42	0.32	0/0/-0.18	0.0	1.1	6.0	4.66	0.30	0/0/-0.23
$\widetilde{GC}$ (P)	0.0	1.4	0.0	1.91	0.29	0/0.17/0	0.0	2.1	0.0	1.92	0.28	0/0.01/0	0.0	2.2	0.0	1.83	0.28	0/0.52/0	0.0	1.1	0.0	1.77	0.27	0/-0.34/0
$\widetilde{GC}$ (O)	0.0	0.0	10.7	1.97	0.64	0/0/-0.35	0.0	0.0	9.2	2.16	0.61	0/0/-0.19	0.0	0.0	8.2	2.50	0.54	0/0/0.18	0.0	0.0	6.8	2.69	0.53	0/0/0.13
$\widetilde{UNIQ}$ (D+P+O)	11.6	1.4	4.4	1.28	0.61	0.06/0.15/-0.52	9.6	2.1	4.0	1.36	0.58	0.28/0.14/-0.40	7.1	2.2	4.4	1.64	0.52	0.62/0.60/-0.02	6.8	1.1	4.2	1.94	0.50	0.53/-0.08/0.12
$\widetilde{UNIQ}$ (D+O)	11.6	0.0	5.5	1.28	0.59	0.08/0/-0.51	9.6	0.0	5.4	1.47	0.55	0.28/0/-0.28	7.1	0.0	6.0	1.84	0.50	0.53/0/0.09	6.8	0.0	5.1	2.12	0.51	0.55/0/0.04
$\widetilde{UNIQ}$ (P+O)	0.0	1.4	9.5	4.06	0.32	0/0/-0.50	0.0	2.1	7.6	3.78	0.33	0/0/-0.53	0.0	2.2	6.5	3.50	0.32	0/0/-0.34	0.0	1.1	6.0	3.26	0.29	0/0/-0.38
$\widetilde{UNIQ}$ (P)	0.0	1.4	0.0	1.25	0.26	0/-0.26/0	0.0	2.1	0.0	1.31	0.26	0/-0.42/0	0.0	2.2	0.0	1.31	0.27	0/-0.19/0	0.0	1.1	0.0	1.31	0.27	0/-0.55/0
$\widetilde{UNIQ}$ (O)	0.0	0.0	10.7	1.80	0.63	0/0/-0.34	0.0	0.0	9.2	2.01	0.61	0/0/-0.16	0.0	0.0	8.2	2.31	0.54	0/0/0.19	0.0	0.0	6.8	2.55	0.52	0/0/0.14

50% of Observability

	Domain Incompleteness 20%						Domain Incompleteness 40%						Domain Incompleteness 60%						Domain Incompleteness 80%					
	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO	$|D|$	$|P|$	$|O|$	$|S|$	F_1	CD/CP/CO
Baseline (h_{gc})	11.5	0.0	0.0	1.36	0.44	0.38/0/0	4.8	0.0	0.0	1.50	0.36	0.43/0/0	3.5	0.0	0.0	1.74	0.31	0.63/0/0	2.9	0.0	0.0	1.95	0.34	0.71/0/0
Baseline (h_{uniq})	11.5	0.0	0.0	1.27	0.45	0.32/0/0	4.8	0.0	0.0	1.48	0.34	0.41/0/0	3.5	0.0	0.0	1.56	0.32	0.31/0/0	2.9	0.0	0.0	1.74	0.32	0.46/0/0
$\widetilde{GC}$ (D+P+O)	11.6	1.4	8.8	1.24	0.76	0.29/0.17/-0.45	9.6	2.1	7.9	1.31	0.76	0.63/0.37/-0.20	7.1	2.2	8.5	1.53	0.70	0.85/0.55/0.24	6.8	1.1	8.2	1.76	0.67	0.65/0.05/0.35
$\widetilde{GC}$ (D+O)	11.6	0.0	10.9	1.26	0.76	0.34/0.0/-0.35	9.6	0.0	10.6	1.33	0.76	0.66/0.0/-0.04	7.1	0.0	11.4	1.59	0.69	0.84/0.0/0.46	6.8	0.0	9.8	1.84	0.67	0.65/0.0/0.39
$\widetilde{GC}$ (P+O)	0.0	1.4	18.6	5.19	0.18	0/-0.11/-0.37	0.0	2.1	14.9	4.73	0.20	0/-0.04/-0.28	0.0	2.2	12.6	4.82	0.19	0/0.34/-0.17	0.0	1.1	11.7	4.87	0.16	0/-0.50/-0.28
$\widetilde{GC}$ (P)	0.0	1.4	0.0	1.94	0.31	0/0.20/0	0.0	2.1	0.0	1.96	0.35	0/0.11/0	0.0	2.2	0.0	2.20	0.33	0/0.65/0	0.0	1.1	0.0	1.70	0.30	0/-0.27/0
$\widetilde{GC}$ (O)	0.0	0.0	20.7	1.54	0.74	0/0/-0.07	0.0	0.0	17.8	1.59	0.74	0/0/0.10	0.0	0.0	15.7	1.88	0.66	0/0/0.45	0.0	0.0	13.3	2.08	0.64	0/0/0.31
$\widetilde{UNIQ}$ (D+P+O)	11.6	1.4	8.8	1.13	0.72	0.32/0.26/-0.32	9.6	2.1	7.9	1.18	0.70	0.33/0.37/-0.07	7.1	2.2	8.5	1.38	0.64	0.83/0.64/0.36	6.8	1.1	8.2	1.60	0.62	0.63/0.06/0.40
$\widetilde{UNIQ}$ (D+O)	11.6	0.0	10.9	1.14	0.70	0.39/0.0/-0.21	9.6	0.0	10.6	1.25	0.67	0.45/0.0/0.17	7.1	0.0	11.4	1.52	0.62	0.68/0.0/0.44	6.8	0.0	9.8	1.78	0.63	0.66/0.0/0.33
$\widetilde{UNIQ}$ (P+O)	0.0	1.4	18.6	4.02	0.32	0/-0.13/-0.48	0.0	2.1	14.9	3.83	0.35	0/-0.17/-0.51	0.0	2.2	12.6	3.72	0.33	0/0.18/-0.29	0.0	1.1	11.7	3.21	0.29	0/-0.57/-0.41
$\widetilde{UNIQ}$ (P)	0.0	1.4	0.0	1.25	0.26	0/-0.26/0	0.0	2.1	0.0	1.31	0.27	0/-0.42/0	0.0	2.2	0.0	1.31	0.27	0/-0.19/0	0.0	1.1	0.0	1.31	0.27	0/-0.55/0
$\widetilde{UNIQ}$ (O)	0.0	0.0	20.7	1.39	0.74	0/0/-0.02	0.0	0.0	17.8	1.46	0.74	0/0/0.11	0.0	0.0	15.7	1.68	0.66	0/0/0.46	0.0	0.0	13.3	1.93	0.64	0/0/0.32

Table APPENDIX B.3: Performance results for ablation study, comparing the baseline approaches against our heuristics using some possible combinations of landmark types, separated by observability (10%, 30%, and 50%).

70% of Observability

	Domain Incompleteness 20%						Domain Incompleteness 40%						Domain Incompleteness 60%						Domain Incompleteness 80%					
	$\lvert D\rvert$	$\lvert P\rvert$	$\lvert O\rvert$	$\lvert S\rvert$	$\lvert F_1\rvert$	CD/CP/CO	$\lvert D\rvert$	$\lvert P\rvert$	$\lvert O\rvert$	$\lvert S\rvert$	$\lvert F_1\rvert$	CD/CP/CO	$\lvert D\rvert$	$\lvert P\rvert$	$\lvert O\rvert$	$\lvert S\rvert$	$\lvert F_1\rvert$	CD/CP/CO	$\lvert D\rvert$	$\lvert P\rvert$	$\lvert O\rvert$	$\lvert S\rvert$	$\lvert F_1\rvert$	CD/CP/CO
Baseline (h_{gc})	11.5	0.0	0.0	1.35	0.50	0.49/0/0	4.8	0.0	0.0	1.46	0.41	0.54/0/0	3.5	0.0	0.0	1.70	0.33	0.67/0/0	2.9	0.0	0.0	1.90	0.37	0.71/0/0
Baseline (h_{uniq})	11.6	0.0	0.0	1.23	0.49	0.45/0/0	4.8	0.0	0.0	1.37	0.36	0.49/0/0	3.5	0.0	0.0	1.45	0.33	0.42/0/0	2.9	0.0	0.0	1.67	0.34	0.50/0/0
$\widetilde{GC}$ (D+P+O)	11.5	1.4	14.4	1.09	0.88	0.38/0.27/-0.14	9.6	2.1	12.8	1.14	0.87	0.62/0.31/0.12	7.1	2.2	13.6	1.30	0.83	0.72/0.32/0.27	6.8	1.1	13.2	1.43	0.80	0.65/0.11/0.48
$\widetilde{GC}$ (D+O)	11.6	0.0	17.5	1.11	0.87	0.38/0/-0.04	9.6	0.0	17.0	1.15	0.87	0.62/0/0.20	7.1	0.0	18.0	1.33	0.82	0.78/0/0.51	6.8	0.0	15.6	1.48	0.80	0.66/0/0.48
$\widetilde{GC}$ (P+O)	0.0	1.4	29.7	5.62	0.31	0/-0.09/-0.40	0.0	2.1	24.1	5.03	0.35	0/-0.04/-0.29	0.0	2.2	20.4	5.13	0.35	0/0.25/-0.25	0.0	1.1	19.0	5.11	0.33	0/-0.48/-0.34
$\widetilde{GC}$ (P)	0.0	1.4	0.0	2.10	0.30	0/0.26/0	0.0	2.1	0.0	2.07	0.31	0/0.10/0	0.0	2.2	0.0	2.19	0.28	0/0.59/0	0.0	1.1	0.0	1.72	0.27	0/-0.21/0
$\widetilde{GC}$ (O)	0.0	0.0	33.0	1.25	0.84	0/0/0.038	0.0	0.0	28.5	1.28	0.84	0/0/0.30	0.0	0.0	24.9	1.47	0.78	0/0/0.52	0.0	0.0	21.4	1.63	0.77	0/0/0.47
$\widetilde{UNIQ}$ (D+P+O)	11.6	1.4	14.4	1.09	0.83	0.46/0.37/-0.09	9.6	2.1	12.8	1.10	0.80	0.35/0.34/0.21	7.1	2.2	13.6	1.23	0.76	0.84/0.56/0.46	6.8	1.1	13.2	1.37	0.74	0.64/0.18/0.56
$\widetilde{UNIQ}$ (D+O)	11.6	0.0	17.5	1.09	0.81	0.61/0/0.17	9.6	0.0	17.0	1.15	0.77	0.45/0/0.46	7.1	0.0	18.0	1.32	0.75	0.70/0/0.57	6.8	0.0	15.6	1.46	0.76	0.68/0/0.52
$\widetilde{UNIQ}$ (P+O)	0.0	1.4	29.7	4.20	0.32	0/-0.09/-0.49	0.0	2.1	24.1	3.97	0.35	0/-0.14/-0.51	0.0	2.2	20.4	3.91	0.34	0/0.18/-0.31	0.0	1.1	19.0	3.13	0.29	0/-0.56/-0.46
$\widetilde{UNIQ}$ (P)	0.0	1.4	0.0	1.25	0.26	0/-0.27/0	0.0	2.1	0.0	1.31	0.27	0/-0.42/0	0.0	2.2	0.0	1.31	0.27	0/-0.20/0	0.0	1.1	0.0	1.31	0.27	0/-0.55/0
$\widetilde{UNIQ}$ (O)	0.0	0.0	33.0	1.14	0.83	0/0/0.16	0.0	0.0	28.5	1.19	0.83	0/0/0.26	0.0	0.0	24.9	1.35	0.77	0/0/0.49	0.0	0.0	21.4	1.54	0.75	0/0/0.46

100% of Observability

	Domain Incompleteness 20%						Domain Incompleteness 40%						Domain Incompleteness 60%						Domain Incompleteness 80%					
	$\lvert D\rvert$	$\lvert P\rvert$	$\lvert O\rvert$	$\lvert S\rvert$	$\lvert F_1\rvert$	CD/CP/CO	$\lvert D\rvert$	$\lvert P\rvert$	$\lvert O\rvert$	$\lvert S\rvert$	$\lvert F_1\rvert$	CD/CP/CO	$\lvert D\rvert$	$\lvert P\rvert$	$\lvert O\rvert$	$\lvert S\rvert$	$\lvert F_1\rvert$	CD/CP/CO	$\lvert D\rvert$	$\lvert P\rvert$	$\lvert O\rvert$	$\lvert S\rvert$	$\lvert F_1\rvert$	CD/CP/CO
Baseline (h_{gc})	11.5	0.0	0.0	1.42	0.53	0.58/0/0	4.8	0.0	0.0	1.57	0.42	0.62/0/0	3.5	0.0	0.0	1.77	0.36	0.66/0/0	2.9	0.0	0.0	1.89	0.38	0.72/0/0
Baseline (h_{uniq})	11.5	0.0	0.0	1.39	0.53	0.59/0/0	4.8	0.0	0.0	1.54	0.38	0.54/0/0	3.5	0.0	0.0	1.63	0.35	0.48/0/0	2.9	0.0	0.0	1.83	0.35	0.52/0/0
$\widetilde{GC}$ (D+P+O)	11.6	1.4	21.2	1.04	0.96	0.35/0.23/0.32	9.6	2.1	19.0	1.07	0.94	0.46/0.31/0.34	7.1	2.2	19.6	1.21	0.91	0.58/0.23/0.41	6.8	1.1	19.1	1.27	0.88	0.57/0.25/0.60
$\widetilde{GC}$ (D+O)	11.6	0.0	25.5	1.05	0.95	0.35/0/0.30	9.6	0.0	24.7	1.08	0.93	0.46/0/0.36	7.1	0.0	25.7	1.23	0.90	0.61/0/0.53	6.8	0.0	22.5	1.29	0.88	0.57/0/0.60
$\widetilde{GC}$ (P+O)	0.0	1.4	42.8	6.10	0.32	0/-0.09/-0.47	0.0	2.1	35.2	5.47	0.36	0/-0.03/-0.37	0.0	2.2	29.7	5.39	0.35	0/0.27/-0.23	0.0	1.1	27.8	5.44	0.32	0/-0.46/-0.40
$\widetilde{GC}$ (P)	0.0	1.4	0.0	2.46	0.24	0/0.29/0	0.0	2.1	0.0	2.38	0.23	0/0.12/0	0.0	2.2	0.0	2.32	0.23	0/0.58/0	0.0	1.1	0.0	1.88	0.22	0/-0.17/0
$\widetilde{GC}$ (O)	0.0	0.0	47.4	1.12	0.91	0/0/0.12	0.0	0.0	41.4	1.13	0.90	0/0/0.39	0.0	0.0	35.9	1.28	0.86	0/0/0.57	0.0	0.0	31.2	1.39	0.84	0/0/0.58
$\widetilde{UNIQ}$ (D+P+O)	11.6	1.4	21.2	1.01	0.92	0.36/0.37/0.31	9.6	2.1	19.0	1.03	0.88	0.15/0.25/0.49	7.1	2.2	19.6	1.09	0.85	0.65/0.39/0.56	6.8	1.1	19.1	1.17	0.83	0.53/0.26/0.65
$\widetilde{UNIQ}$ (D+O)	11.6	0.0	25.5	1.02	0.90	0.46/0/0.53	9.6	0.0	24.7	1.07	0.87	0.22/0/0.62	7.1	0.0	25.7	1.16	0.85	0.58/0/0.59	6.8	0.0	22.5	1.23	0.85	0.55/0/0.60
$\widetilde{UNIQ}$ (P+O)	0.0	1.4	42.8	5.40	0.33	0/-0.07/-0.50	0.0	2.1	35.2	5.16	0.36	0/-0.12/-0.51	0.0	2.2	29.7	5.06	0.34	0/0.18/-0.29	0.0	1.1	27.8	5.01	0.30	0/-0.54/-0.54
$\widetilde{UNIQ}$ (P)	0.0	1.4	0.0	2.22	0.26	0/-0.27/0	0.0	2.1	0.0	2.25	0.27	0/-0.42/0	0.0	2.2	0.0	2.30	0.27	0/-0.19/0	0.0	1.1	0.0	2.31	0.27	0/-0.55/0
$\widetilde{UNIQ}$ (O)	0.0	0.0	47.4	1.05	0.89	0/0/0.33	0.0	0.0	41.4	1.09	0.90	0/0/0.32	0.0	0.0	35.9	1.23	0.85	0/0/0.51	0.0	0.0	31.2	1.35	0.82	0/0/0.53

Table APPENDIX B.4: Performance results for ablation study, comparing the baseline approaches against our heuristics using some possible combinations of landmark types, separated by observability (70% and 100%).

APPENDIX C – GOAL RECOGNITION OVER NOMINAL MODELS - DETAILED RESULTS

In this Appendix, we present detailed experimental results of our approaches for goal recognition over *nominal models*. Specifically, we report on a set of experiments that aims to evaluate the performance of our recognition approaches ηMIRRORING and $\Delta(Obs, G)$, presented in Chapter 4, over both *actual* and *nominal models*, when dealing with *linear* and *non–linear* domain models.

Linear Domain Models

Tables APPENDIX C.1, APPENDIX C.2, and APPENDIX C.3 show the experimental results for three (linear) LQR–based domain models (formally defined in Section 4.6.1): 1D–LQR–Navigation, 2D–LQR–Navigation with a single vehicle, and 2D–LQR–Navigation with multiple vehicles. We measure the recognition performance of our approaches over these domains in three settings: *online* (the recognition process is performed incrementally after observing a state), *offline* (considering observed states at once), *1st observation*. For evaluation, we use the same metrics we used in Section 4.6.4, namely, True Positive Rate (TPR), False Positive Rate (FPR), *Top-2*. In these tables, the column M represents the model type (A represents *actual models*, and N represents *nominal models*), % *Obs* is observation level, and N is the total number of observed states. Note that the average number of goal hypothesis $|\mathcal{G}|$ in the datasets for these three domains is 5. We use two levels of observability in these sets of experiments: 5% and 10%, more specifically, for each of these levels of observability we observe 5 and 10 states per problem.

Though the results are very similar when comparing our approaches for all settings in all these three linear domains, we note that, from the results in these tables, the best results we had are for *offline* goal recognition when using the ηMIRRORING approach. The 1D–LQR–Navigation domain is the linear domain in which our recognition approaches have achieved better results, and we note that it might due to the complexity of this domain, in which we have only one dimension and one vehicle. After analyzing the results of $\Delta(Obs, G)$, we note that this approach has achieved relatively poor results for all linear domains due to the fact the planner we used most likely get trapped in local minimum for most problems in these datasets.

As we mentioned in Section 4.6.4, we dug deeper into the extracted rewards for J^+ and J^-, and to overcome this issue, we aim to use different solvers to see how they will behave when dealing with the modified cost functions of $\Delta(Obs, G)$.

1D–LQR–Navigation (Linear Domain)												
				Online				Offline			1st Observation	
Approach	M	Obs (%)	N	Top-2	TPR	FPR	N	Top-2	TPR	FPR	Top-2	TPR	FPR
ηMirroring	A	5	150	0.87	0.73	0.06	30	1.00	0.97	0.01	0.77	0.40	0.13
$\Delta(Obs, G)$	A	5	150	0.56	0.29	0.16	30	0.63	0.47	0.12	0.40	0.20	0.18
ηMirroring	A	10	300	0.90	0.76	0.05	30	1.00	0.97	0.01	0.63	0.33	0.15
$\Delta(Obs, G)$	A	10	300	0.45	0.26	0.17	30	0.47	0.27	0.16	0.37	0.27	0.16
ηMirroring	N	5	150	0.72	0.54	0.11	30	0.93	0.80	0.05	0.53	0.27	0.17
$\Delta(Obs, G)$	N	5	150	0.57	0.29	0.16	30	0.57	0.33	0.15	0.77	0.33	0.15
ηMirroring	N	10	300	0.77	0.51	0.11	30	0.93	0.83	0.05	0.37	0.20	0.17
$\Delta(Obs, G)$	N	10	300	0.53	0.32	0.15	30	0.43	0.27	0.16	0.63	0.40	0.13

Table APPENDIX C.1: Experimental results of our recognition approaches ηMirroring and $\Delta(Obs, G)$, over both *actual* and *nominal models* for the 1D–LQR–Navigation linear domain.

2D–LQR–Navigation Single Vehicle (Linear Domain)												
				Online				Offline			1st Observation	
Approach	M	Obs (%)	N	Top-2	TPR	FPR	N	Top-2	TPR	FPR	Top-2	TPR	FPR
ηMirroring	A	5	150	0.89	0.86	0.03	30	1.00	1.00	0.00	0.60	0.50	0.10
$\Delta(Obs, G)$	A	5	150	0.47	0.25	0.15	30	0.37	0.23	0.16	0.53	0.30	0.14
ηMirroring	A	10	300	0.92	0.86	0.03	30	1.00	1.00	0.00	0.63	0.40	0.12
$\Delta(Obs, G)$	A	10	300	0.42	0.27	0.15	30	0.30	0.20	0.16	0.57	0.33	0.13
ηMirroring	N	5	150	0.64	0.41	0.12	30	0.73	0.63	0.07	0.50	0.27	0.15
$\Delta(Obs, G)$	N	5	150	0.43	0.19	0.17	30	0.37	0.22	0.16	0.50	0.13	0.18
ηMirroring	N	10	300	0.62	0.43	0.11	30	0.73	0.57	0.09	0.43	0.23	0.15
$\Delta(Obs, G)$	N	10	300	0.45	0.24	0.16	30	0.43	0.17	0.17	0.47	0.25	0.16

Table APPENDIX C.2: Experimental results of our recognition approaches ηMirroring and $\Delta(Obs, G)$, over both *actual* and *nominal models* for the 2D–LQR–Navigation single vehicle linear domain.

2D–LQR–Navigation with Multiple Vehicles (Linear Domain)														
				ONLINE				OFFLINE			1ST OBSERVATION		
Approach	*M*	*Obs* (%)	*N*	*Top-2*	*TPR*	*FPR*	*N*	*Top-2*	*TPR*	*FPR*	*Top-2*	*TPR*	*FPR*
ηMIRRORING	A	5	150	0.84	0.71	0.06	30	0.90	0.83	0.03	0.63	0.43	0.12
$\Delta(Obs, G)$	A	5	150	0.45	0.18	0.18	30	0.47	0.20	0.18	0.53	0.27	0.16
ηMIRRORING	A	10	300	0.88	0.73	0.06	30	0.93	0.90	0.02	0.67	0.23	0.17
$\Delta(Obs, G)$	A	10	300	0.49	0.26	0.16	30	0.57	0.37	0.14	0.47	0.23	0.17
ηMIRRORING	N	5	150	0.61	0.43	0.12	30	0.83	0.57	0.09	0.30	0.20	0.18
$\Delta(Obs, G)$	N	5	150	0.34	0.16	0.19	30	0.37	0.23	0.17	0.27	0.08	0.21
ηMIRRORING	N	10	300	0.74	0.54	0.10	30	0.93	0.77	0.05	0.43	0.33	0.15
$\Delta(Obs, G)$	N	10	300	0.37	0.21	0.18	30	0.40	0.18	0.18	0.47	0.30	0.17

Table APPENDIX C.3: Experimental results of our recognition approaches ηMIRRORING and $\Delta(Obs, G)$, over both *actual* and *nominal models* for the 2D–LQR–Navigation linear domain with multiple vehicles.

Non-Linear Domain Models

Tables APPENDIX C.4 and APPENDIX C.5 show the experimental results for two (non–linear) navigation domains models (formally defined in Section 4.6.1): 2D–NAV and 3D–NAV. For these sets of experiments, we used four levels of observability: 5%, 10%, 30%, and 50%. Since the planning horizon H for these two domains is 20, for 5% of observability we have one observed state, whereas for 10% we have two states, and 6 and 10 states for the other two levels of observability, respectively, 30% and 50%. To evaluate the performance of our recognition approaches over non–linear domains, we used the same metrics over the same settings, as we mentioned in the previous section. Note that, our recognition approaches performed better over non–linear domains for all evaluated settings, in comparison to the results for linear domains, especially for offline goal recognition.

2D–NAV (Non-Linear Domain)														
				ONLINE				OFFLINE				1ST OBSERVATION		
Approach	*M*	*Obs* (%)	*N*	*Top-2*	*TPR*	*FPR*	*N*	*Top-2*	*TPR*	*FPR*	*Top-2*	*TPR*	*FPR*
ηMIRRORING	A	5	10	1.00	1.00	0.00	10	1.00	1.00	0.00	1.00	1.00	0.00
$\Delta(Obs, G)$	A	5	10	0.70	0.60	0.10	10	0.70	0.60	0.10	0.70	0.60	0.10
ηMIRRORING	A	10	20	1.00	0.75	0.06	10	1.00	0.80	0.05	1.00	0.70	0.07
$\Delta(Obs, G)$	A	10	20	0.50	0.30	0.17	10	0.50	0.20	0.20	0.50	0.40	0.15
ηMIRRORING	A	30	60	0.70	0.47	0.13	10	0.80	0.50	0.12	0.50	0.30	0.17
$\Delta(Obs, G)$	A	30	60	0.58	0.35	0.16	10	0.60	0.40	0.15	0.60	0.40	0.15
ηMIRRORING	A	50	100	0.96	0.86	0.04	10	1.00	1.00	0.00	0.80	0.40	0.15
$\Delta(Obs, G)$	A	50	100	0.60	0.35	0.16	10	0.70	0.50	0.12	0.50	0.20	0.20
ηMIRRORING	N	5	10	0.90	0.80	0.05	10	0.90	0.80	0.05	0.90	0.80	0.05
$\Delta(Obs, G)$	N	5	10	0.60	0.20	0.20	10	0.60	0.20	0.20	0.60	0.20	0.20
ηMIRRORING	N	10	20	0.80	0.60	0.10	10	0.80	0.70	0.07	0.80	0.50	0.12
$\Delta(Obs, G)$	N	10	20	0.45	0.25	0.19	10	0.50	0.30	0.17	0.40	0.20	0.20
ηMIRRORING	N	30	60	0.77	0.62	0.10	10	1.00	1.00	0.00	0.50	0.20	0.20
$\Delta(Obs, G)$	N	30	60	0.48	0.10	0.23	10	0.50	0.10	0.23	0.50	0.10	0.23
ηMIRRORING	N	50	100	0.87	0.68	0.08	10	1.00	1.00	0.00	0.50	0.20	0.20
$\Delta(Obs, G)$	N	50	100	0.42	0.14	0.21	10	0.50	0.30	0.17	0.40	0.20	0.20

Table APPENDIX C.4: Experimental results of our recognition approaches over both *actual* and *nominal models* for the 2D–NAV non-linear domain.

3D–NAV (Non-Linear Domain)													
				ONLINE				OFFLINE			1ST OBSERVATION		
Approach	*M*	*Obs* (%)	*N*	*Top-2*	*TPR*	*FPR*	*N*	*Top-2*	*TPR*	*FPR*	*Top-2*	*TPR*	*FPR*
ηMIRRORING	A	5	10	0.70	0.70	0.07	10	0.70	0.70	0.07	0.70	0.70	0.07
$\Delta(Obs, G)$	A	5	10	0.90	0.60	0.10	10	0.90	0.60	0.10	0.90	0.60	0.10
ηMIRRORING	A	10	20	0.80	0.80	0.05	10	1.00	1.00	0.00	0.60	0.60	0.10
$\Delta(Obs, G)$	A	10	20	0.65	0.35	0.16	10	0.70	0.50	0.12	0.60	0.20	0.20
ηMIRRORING	A	30	60	0.85	0.63	0.09	10	1.00	1.00	0.00	0.70	0.40	0.15
$\Delta(Obs, G)$	A	30	60	0.72	0.38	0.15	10	0.90	0.60	0.10	0.50	0.10	0.23
ηMIRRORING	A	50	100	0.79	0.60	0.10	10	1.00	1.00	0.00	0.70	0.30	0.17
$\Delta(Obs, G)$	A	50	100	0.65	0.39	0.15	10	0.90	0.80	0.05	0.40	0.10	0.23
ηMIRRORING	N	5	10	0.80	0.60	0.10	10	0.80	0.60	0.10	0.80	0.60	0.10
$\Delta(Obs, G)$	N	5	10	0.90	0.70	0.07	10	0.90	0.70	0.07	0.90	0.70	0.07
ηMIRRORING	N	10	20	0.65	0.50	0.12	10	0.80	0.60	0.10	0.50	0.40	0.15
$\Delta(Obs, G)$	N	10	20	0.60	0.45	0.14	10	0.60	0.40	0.15	0.60	0.50	0.12
ηMIRRORING	N	30	60	0.75	0.60	0.10	10	1.00	1.00	0.00	0.50	0.30	0.17
$\Delta(Obs, G)$	N	30	60	0.52	0.30	0.17	10	0.50	0.30	0.17	0.50	0.30	0.17
ηMIRRORING	N	50	100	0.76	0.61	0.10	10	1.00	1.00	0.00	0.50	0.30	0.17
$\Delta(Obs, G)$	N	50	100	0.54	0.29	0.18	10	1.00	0.30	0.17	0.50	0.30	0.17

Table APPENDIX C.5: Experimental results of our recognition approaches ηMIRRORING and $\Delta(Obs, G)$, over both *actual* and *nominal models* for the 3D–NAV non-linear domain.