UMEÅ UNIVERSITY

Computational Models for Intent Recognition in Robotic Systems

Michele Persiani

Licentiate Thesis
Department of Computing Science
Umeå University, Sweden
7 September 2020

Department of Computing Science
Umeå University
SE-901 87 Umeå, Sweden

michelep@cs.umu.se

ISBN 978-91-7855-330-3
ISSN 0348-0542
UMINF 20.8

Printed by UmU Print Service, Umeå University, 2020

Abstract

The ability to infer and mediate intentions has been recognized as a crucial task in recent robotics research, where it is agreed that robots are required to be equipped with intentional mechanisms in order to participate in collaborative tasks with humans.

Reasoning about—or rather, perceiving—intentions enables robots to infer what other agents are doing, to communicate what are their plans, or to take proactive decisions. Intent recognition relates to several system requirements, such as the need of an enhanced collaboration mechanism in human-machine interactions, the need for adversarial technology in competitive scenarios, ambient intelligence, or predictive security systems.

When attempting to describe what an intention is, agreement exists to represent it as a plan together with the goal it attempts to achieve. Being compatible with computer science concepts, this representation enables to handle intentions with methodologies based on planning, such as the Planning Domain Description Language or Hierarchical Task Networks.

In this licentiate we describe how intentions can be processed using classical planning methods, with an eye also on newer technologies such as deep networks. Our goal is to study and define computational models that would allow robotic agents to infer, construct and mediate intentions. Additionally, we explore how intentions in the form of abstract plans can be grounded to sensorial data, and in particular we provide discussion on grounding over speech utterances and affordances, that correspond to the action possibilities offered by an environment.

Sammanfattning

Förmågan att prediktera och kommunicera avsikter ses som en viktig uppgift i modern robotforskning, där man inser att robotar måste vara utrustade med sådana mekanismer för att samarbeta med människor.

Att resonera om - eller snarare, att förstå—avsikter gör det möjligt för robotar att sluta sig till vad andra agenter gör, kommunicera vad som är deras planer, eller fatta proaktiva beslut. Detta är relaterat till flera systemkrav, såsom behovet av en förbättrad mekanism för interaktion mellan människor och maskiner, behovet av "adversarial technology" i konkurrensscenarier, "ambient intelligens" eller predikterande säkerhetssystem.

När man försöker beskriva vad en avsikt är, råder det enighet om att representera den som en plan tillsammans med det mål den försöker uppnå. Eftersom det är kompatibelt med datavetenskapliga koncept, möjliggör denna representation att avsikter hanteras med metodologier baserade på planering, såsom "Planning Domain Description Language" eller "Hierarchical Task Networks".

I denna licentiatavhandling beskriver vi hur avsikter kan behandlas med klassiska planeringsmetoder, men även med nyare teknik som djupa nätverk. Vårt mål är att studera och definiera beräkningsmodeller som kan göra det möjligt för robot-agenter att prediktera, konstruera, och förmedla avsikter. Dessutom undersöker vi hur avsikter i form av abstrakta planer kan baseras på sensordata, och i synnerhet diskuterar vi hur "grounding" av tal och "affordances" motsvarar de handlingar som är möjliga i en given miljö.

Acknowledgements

Finally licentitate has arrived. Even if it marks only a middle milestone of the PhD it already feels like an incredible achievement, which was absolutely impossible for me to reach without the help I've been receiving from the many persons who have continuosly been supporting me.
The first person I would like to thank is Maitreyee Tewari, especially for all of her enormous patience. Dealing with a person always busy with his own work is not easy; and she's the only one I know to be this incredibly enduring. Thanks, and keep up the good work for your own PhD!
—The second person for whom my sincere acknowledgements are is Thomas Hellström. We had so many interesting discussions and from my side it was such an intense relationship. We can undoubtedly say that if I'm here it's because of him and his support.
Following, I thank all of the rest of the Intelligent Robotics group; with Ola, Peter, Suna, Ahmad, Neha, Avinash, Adna. We had very nice afterworks and board games sessions. Beware of them, because I never encountered—such competitive board games players!
I would like to thank all of the professors, researchers and PhD students I met and am meeting along my journey. With some of them I bonded more, some of them less; it is though always a pleasure to feel part of a vibrant and striving community.
Last but definitely not least, my family, Mamma, Papá, Cinno, Cinna and all of my friends for their continuous support, both psychological when in trouble and material. Ghedo, Jack, Franz, Farolf, the amounts of DotA nights we're having is incredible and that is absolutely appreciated. It will come—the time where I can play Enigma mid-lane, just wait!

Preface

This thesis contains an introduction to intention and computational models for intention recognition, which are here mainly discussed by classical planning methods. We further provide discussion on how to ground intentions in affordances and sensory data such as speech. The following papers are also included at the end of the licentiate.

Paper I — *Michele Persiani, Thomas Hellström.* **Intent Recognition From Speech and Plan Recognition**. *Proceedings of the 18th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS), pp. 212-223, 2020.*

Paper II — *Michele Persiani, Thomas Hellström.* **Unsupervised Inference of Object Affordance from Text Corpora**. *Proceedings of the 22nd Nordic Conference on Computational Linguistics (NoDaLiDa), pp. 115-120, 2019.*

Paper III — *Michele Persiani, Çağatay Odabaşı, Florenz Graf, Mohit Kalra, Thomas Hellström, Birgit Graf.* **Traveling Drinksman—A Mobile Service Robot for People in Care-Homes**. *Accepted at the 52nd International Symposium of Robotics (ISR), 2020.*

Paper IV — *Michele Persiani, Maitreyee Tewari.* **Extended abstract: Mediating Joint Intentions with a Dialogue Management System**. *Accepted at 1st New Foundations for Human-Centered AI Workshop (NeHuAI), 2020.*

This work has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 721619 for the SOCRATES project.

Semantic map

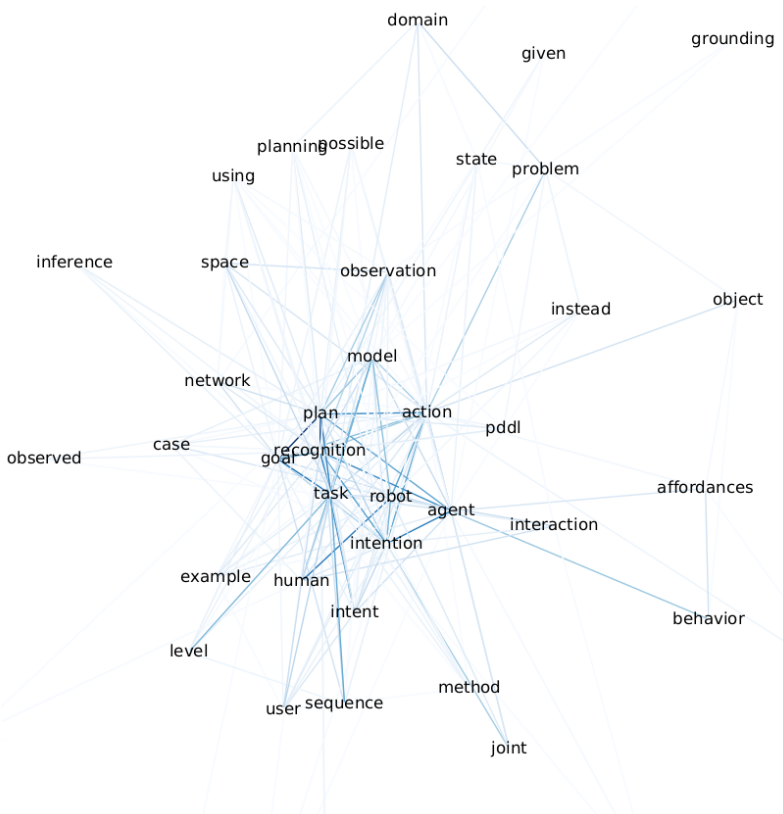

Figure 1: Semantic map of this licentiate. Edges color indicate the degree of co-occurrence of words in the text.

Contents

1 Introduction **1**

2 Intention **5**

3 Intent Recognition **7**
3.1 Intent Recognition through Plan Recognition 9
3.1.1 Plan recognition with PDDL 11
3.1.2 Plan recognition with Hierarchical Task Networks . . 15
3.2 Plan recognition with deep networks 16
3.3 Other recognition methods 18
3.4 Grounding of observations 19
3.4.1 Speech 20
3.4.2 Affordances 22

4 Contributions **25**
4.1 Paper I 25
4.2 Paper II 26
4.3 Paper III 27
4.4 Paper IV 28

5 Future work **29**
5.1 Deep models for goal recognition 30
5.2 Joint Intentions 30
5.3 Model Reconciliation 31

Paper I **39**

Paper II **53**

Paper III **61**

Paper IV **69**

Chapter 1

Introduction

In robotics, intent recognition is the algorithmic task of inferring the intention of an observed agent from a stream of observations. Intent recognition can be grouped together with a series of recognition tasks [28] such as goal recognition, plan recognition, behavior recognition, policy recognition and activity recognition, which model properties and behaviors of agents while relating them to a task they are performing, and can be applied in both single-agent or multi-agent settings.

An intent recognition architecture is a system devoted to infer the intention of the observed agents. Commonly, intent recognition architectures accumulate observations about what the agents are doing, then, the observations are matched with prior gathered knowledge to find what they are trying to achieve, what are their intentions. Depending on the purpose of the system, inferred intentions can correspond to different system actions or can be used to keep an updated mental model of the agents such as in Theory of Mind [9], where the intentional state plays an important role.

Intent recognition can find many applications both in collaborative and competitive settings. In general, systems that are able to predict their users or have a model of them can preemptively suggest relevant options, rather than waiting all of the user's inputs.

In contexts of collaboration between humans and robots, such as in Human-Robot Interaction, knowing what the user is attempting to achieve provides robots a better understanding of the current interaction; it also allows them to mediate goals, to react to their inference or enables multimodal communication. Furthermore, predicting the user can help in main-

taining interaction fluency with the robot [15], which is a desirable property of interactions.

In Human-Robot Teaming instead, successfully recognizing the team's shared intention provides robots the capacity to autonomously participate in the best way, for example by taking charge of a previously unclaimed objective, or by supporting other participants in what they're doing if needed. This is possible by the construction of an "intentional context" to which robots can participate. It is a common case for human-robot teams in scenarios such as rescue scenarios [31, 10], where robots are required to infer what the other agents are doing as part of the team, and hence how they can best collaborate.

In other applications such as in smart-environments, intent recognition is used to support mind control of appliances [33], which can be especially useful for people with bodily or cognitive handicap, such as with the Parkinson disease. Alternatively, smart devices can detect intentions to act autonomously.

In competitive scenarios as in robot soccer, being able to predict the future actions of a player or whole team provides strategic advantage. In security systems, malicious attackers can be identified earlier if intentionality is associated to their early actions.

A main distinction that should be given is between intent recognition and activity recognition [28]. The former is involved in recognizing a goal that the agent will eventually reach in the future, while activity recognition is devoted to identify what the actor is doing in the current moment. Some confusion could arise when considering self-actions together with environmental effects as possible goals for an intention. As an example, the intention of a dancer in stage might be the one of simply dancing, that also corresponds to the activity he's doing. Hence, sometimes the recognized activity could also be interpreted as intention (or viceversa), and for this reason we will refer to intention recognition as the systematic algorithmic capacity of evaluating goals that belong to the future.

In this licentiate we focus on algorithms for intent recognition in the context of robotics; in particular, predictions about intentions are discussed when merging the task execution state with information coming from speech modalities. Together with a broader discussion on the grounding of observations, we further investigate the concept of affordance and how it can allow to specify planning domains which can be used to recognize intentions. We therefore define the following research objectives:

- RO1: How can we leverage computational models from Computer

Science for intent recognition?

- RO2: How to recognize intentions from grounded observations, and in particular from speech?
- RO3: Does the affordance space allow to successfully ground intentions?

The rest of this licentiate is structured as follows. In Chapter 2 we provide brief, relevant background on the origin and definition of intention. In Chapter 3 we introduce intent recognition by a Bayesian formulation and provide some relevant potential applications. In Sections 3.1–3.3 intent recognition is discussed more in detail as implemented by classical planning methods, neural networks or other methods. In Section 3.4 we discuss how intentions can be grounded to observations, with a focus on speech modalities. In Chapter 4 we present the contributions of this licentiate toward the development of intent recognition algorithms. In Chapter 5 we finally provide some potential future direction for my research, in particular to how it can touch adjacent topics such as joint intentions or model reconciliation.

Chapter 2

Intention

Understanding intentions has been recognized as crucial skill in social sciences, as it allows to understand what others are doing and why [29]. Intentionality—thinking in terms of goals, rather than of bodily movements—begins to emerge during children's early years, and not only plays a crucial role during children's infancy, but remains a crucial aspect also in all of later life activities.

Studies show how children begin to understand intentionality during their early years [29, 26, 27], by going through three level of understanding of what an intentional action is: first, the child understands that the action is performed autonomously, rather than being an inanimate action. Secondly, the child is able to perceive the persistence of an actor pursuing his/her goals, that is, the actor recognizes success or failure of his actions. Third, the child perceives that the actor consider and chooses amongst different action plans to achieve the chosen goal.

Intention reading is considered to be a skill which shaped human culture. The natural expertise in perceiving intentions is present when understanding what others are trying to communicate or do, also during cultural exchanges and interactions, that may be structured by a pool of shared symbolic artifacts, such as linguistic symbols. Shared intentionality enables to engage others in co-operative activities containing joint goals, and participates in differentiating human beings from other animals.

Agreement exists in representing intentions as a goal (the desired state to which the actor wants to bring the world to) together with a plan of action to achieve that goal [28, 29]. In fact, this is among the earliest working

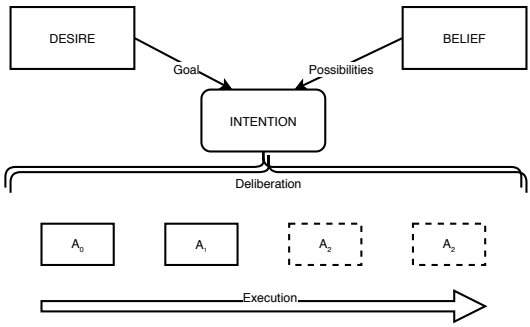

Figure 2.1: Intention in a Belief-Desire-Intention architecture. In order to fulfill a desire, the actor deliberates and commits to an intention.

definitions of intention that was given by Bratman [8]. Intentionality implies commitment, that is, the actor is actively acting out the chosen plan and its actions, rather than just having the plan in mind and not performing it.

A rather strong link exists between intentionality and rational action. When an agent selects and acts a plan achieving a certain goal, we can assume that the plan was selected amongst a set of possible candidates, and that the chosen plan was deemed to be better than the others. Rationality makes the agent to act optimally by not utilizing valuable resources (whatever they may be) more than the necessary.

To better understand how intention is represented in computer science, we will discuss its role in the Belief-Desire-Intention architecture (see Figure 2.1), which has been largely utilized in past research to model artificial agents and how they take and commit to decisions. In BDI models, an agent is represented as having three main components. Its belief system represent the environment in terms of beliefs. These beliefs represent what is true for the agent, which in the general case may not correspond to the ground truth. Alongside, a set of desires guide its behavior. For example, an agent might have the innate desire of not feeling hungry, or of survival. Finally, in order to fulfill a desire given the current belief the agent can instantiate intentions. These intentions correspond to plans of actions, which from its current state allow the agent to fulfill its desires. To maintain consistency between the description from Bratman and BDI architectures, yet without losing generality, we will assume that desires correspond to the possible goals that the agent can be pursuing.

Chapter 3

Intent Recognition

In agreement to the definition given in Chapter 2, we formally define an agent's intention as a goal $\hat{g} \in G$ together with the action plan $\hat{\pi} \in \Pi$ the agent is committed to while pursuing $\hat{g}$ [29]. $\hat{\pi}$ can be either a complete plan or a partial plan directed towards $\hat{g}$ and belongs to a representation level that is abstracted from the raw observations. For example, from a raw video stream of a person walking in a room, the obtained partial plan $\hat{\pi}$ could be the sequence of coordinates (x,y) that the person walked or will walk, $\hat{g}$ the target destination[1].

Given that the observed agent is behaving intentionally, intent recognition is the task of inferring $\hat{g}$ and $\hat{\pi}$ given the (ordered) set of observations $\hat{o}$:

$$\hat{g}, \hat{\pi} = \underset{g \in G, \pi \in \Pi}{\operatorname{argmax}} P(g, \pi | \hat{o}) P(\hat{o}) \tag{3.1}$$

where G and Π are the set of possible goals and the set of possible (partial) plans respectively. $\hat{g}$ and $\hat{\pi}$ are thus the arguments that maximizes the likelyhood of the intent recognition model $P(G, \Pi | O)$.

We additionally introduce a grounding model $P(A|O)$, that from the raw observation space O maps the gathered observations to the abstracted action space A. In this space a plan is an ordered sequence of actions i.e. $\pi = a_0, a_1, ..., a_n$, $\{a_0, a_1, ..., a_n\} \subseteq A, \Pi \subseteq A$. Hence, the raw observations are first grounded to the task space, then, inference of goal and plan is done

[1] For the following discussions we will make the rather strong assumption that observed actions are always intentional. Nevertheless, checking on whether an intention is present is indeed the first step a more complete intent recognition algorithm should do [7].

utilizing that space only. For simplicity we assume for all of the actions to belong to the same abstraction level; a hierarchical decomposition of actions is also possible when using formalisms such as Hierarchical Task Networks (later discussed in Section 3.1.2).

With the introduction of the grounding model a new formulation for intent recognition is therefore obtained as:

$$P(G, \Pi|O) = \sum_{A} P(G, \Pi|A)P(A|O)P(O) \tag{3.2}$$

where the marginalization of A means that the correct grounding is found by averaging all the possible groundings provided by $P(A|O)$. With the additional assumption a goal is independent from the observations having a partial plan $\pi \in \Pi$, the model is further decomposed into:

$$P(G, \Pi|O) = \sum_{A} P(G|\Pi)P(\Pi|A)P(A|O)P(O) \tag{3.3}$$

Therefore, an intent recognition model can be divided into 3 pipelined components: $P(A|O)$ is the grounding model which map raw observations into the task space. $P(\Pi|A)$ is the plan inference component which proposes partial plans extended towards the goals and that best contains the observed actions sequences A. $P(G|\Pi)$ is the goal inference component that from partial plans provides the likelihood of the goals.

This decomposition implies that a partial plan for the agent is first inferred from the abstracted observations, then, the inferred plan is used to infer the agent's goal. Notice that while action sequences $a \in A$ contains only present and past observations, $\hat{\pi}$, the corresponding inferred plan, could contain also future actions, depending on the inner workings of $P(\Pi|A)$. Simple implementations of $P(\Pi|A)$ simply return the input A:

$$P(G|O) = \sum_{A} P(G|A)P(A|O)P(O) \tag{3.4}$$

where no inference on Π is performed and therefore intent recognition becomes equivalent to goal recognition. On the opposite side of the spectrum, when $P(\Pi|A)$ always returns complete plans, the formulation becomes:

$$P(G, \Pi|O) = \sum_{A} P(G)P(\Pi|A)P(A|O)P(O) \tag{3.5}$$

where $P(G)$ is the prior probabilities of the goals. In this case no inference on the goals is needed.

3.1 Intent Recognition through Plan Recognition

The former definition of intention makes particularly easy to frame the inference part of intent recognition into formalisms such as Plan Recognition [11], that is the task of inferring the complete plan (and thereby its goal) of an observed agent.

Plan Recognition algorithms are devoted to find the most likely plan of an agent given a sequence of observations regarding its carried actions, by for example matching the observations to a plan library Π i.e. a dataset of plans. Figure 3.1 depicts the architecture for plan recognition in this setting.

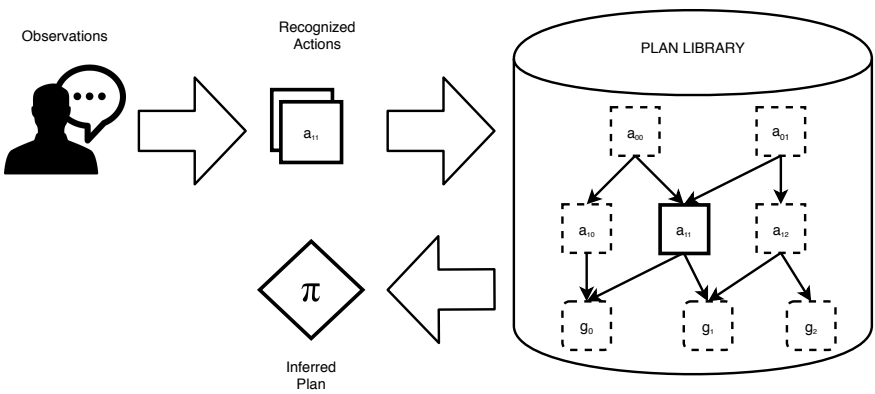

Figure 3.1: Plan recognition using a plan library. Observations are firstly categorized into recognizable actions (i.e. grounded to the task domain), then matched against the plans contained in the library. π, the inferred complete plan, is the plan belonging to the library that best matches the observed actions.

Plan recognition is a case of abductive reasoning i.e. reasoning to best explanation. Reasoning is about the question "Given the observed actions $\hat{a}$, which is the plan in the plan library that best explains them?". In the Bayesian framework this problem can be formulated as:

$$\hat{\pi} = \underset{\pi \in \Pi}{\operatorname{argmax}} P(\pi|\hat{a})P(\hat{a}) \tag{3.6}$$

By applying the Bayes rule on Eq. 3.6:

$$\hat{\pi} = \underset{\pi \in \Pi}{\operatorname{argmax}} P(\hat{a}|\pi)P(\pi) \tag{3.7}$$

hence, the problem is equivalent to finding the plan in the library that best matches the observed actions.

Plan libraries can be either static or dynamic. Static libraries are composed by a fixed dataset of plans, dynamic libraries are instead generated through the utilization of a planner.

As shown in [25], the degree of matching between the observed sequence of actions $\hat{a} \in A$ and the plans $\pi \in \Pi$ towards goals $g \in G$ can be indicated as how costly it is to deviate from an optimal plan achieving g for compliance with the action sequence $\hat{a}$:

$$\forall g \in G \; R(\hat{a}, g) = C(\hat{a}, g) - C(\emptyset, g) \tag{3.8}$$

$C(\hat{a}, g)$ is the cost of achieving a goal g by the means of a plan containing actions $\hat{a}$, while $C(\emptyset, g)$ is the cost of achieving g by the means of an optimal plan. The larger $R(\hat{a}, g)$ is, the more sub-optimal it is to achieve g by performing first $\hat{a}$. $R(\hat{a}, g)$ thus plays the role of a rationality index of actions $\hat{a}$ towards the goals G, or symmetrically, of deviation from the most rational behavior with cost $C(\emptyset, g)$ and expressed by an optimal plan.

Testing Eq. 3.8 over the possible goals provides the degree of optimality of $\hat{a}$ over them. In this setting, we can therefore conclude that the goal being pursued is the one that has the minimum deviation from an optimal plan i.e. which pursuit is being most rational. A probability distribution over the possible goals can be computed as a Boltzmann distribution:

$$\forall g \in G \; P(g|\hat{a}) = \frac{e^{-\beta R(\hat{a}, g)}}{\sum_{g_k \in G} e^{-\beta R(\hat{a}, g_k)}}, \beta > 0 \tag{3.9}$$

where β is the temperature parameter. As inferred goal $\hat{g}$ we can take the goal that has greater likelihood:

$$\hat{g} = \underset{g \in G}{\operatorname{argmax}} P(g|\hat{a}) \tag{3.10}$$

with the inferred complete plan $\hat{\pi}$ being the plan that achieves $\hat{g}$ while containing $\hat{a}$ and that has cost $C(\hat{a}, \hat{g})$. For both dynamic and static libraries finding $\hat{g}$ and $\hat{\pi}$ requires $2|G|$ queries, as for every possible goal both $C(\hat{a}, g)$ and $C(\emptyset, g)$ must be computed. These calls are much more expensive if the

library is dynamic, as the plans computed on the fly. On the opposite, static libraries are faster but allow to evaluate only the plans in it contained, which can be a constraining situation in the case of large state-spaces.

Due to the above discussion, plan recognition encapsulates the *principle of rational action*, that is it assumes the agent is rational when pursuing its goals. Rationality is expressed in terms of preference towards optimality: having a goal $\hat{g}$, the agent will attempt to take the least costly plan to fulfill it. In real scenarios, for the agents to be rational will always be an hypothesis. Nevertheless, such an hypothesis is necessary to associate to the streams of actions the candidate directions they've been pursuing, over which a deviation can be measured.

Several properties are desirable for plan recognition algorithms [28]. Some relevant are **Speed**: inference should be fast and possibly online relative to the time actions require. **Precision**: predictions should have high degrees of confidence. **Earliness**: accurate predictions should be available as early as possible. **Partial prediction**: when full predictions of complete plans are not sufficiently reliable, partial prediction about the agent's plan, or action schema, can instead be provided.

As also described in the next sections, two main classes of plan recognition algorithms are distinguishable, hierarchical and non-hierarchical. Hierarchical recognition recursively groups sequences of low-level action into a single higher level task. Non-hierarchical maintain instead a flat action schema.

3.1.1 Plan recognition with PDDL

Plan Recognition as it is discussed in the previous section can be realized using the Planning Domain Description Language (PDDL). PDDL [21] is a standard language to specify planning domains for what is usually referred to as classical planning. It is based on the STRIPS syntax and uses predicate logic to describe the current system state. PDDL was first proposed in the year 1998 and since then it has been incrementally extended. Every version extends the previous one with functionalities that allow to solve problems of varying complexity, in terms of what is possible to describe. A hierarchical decomposition of actions is not directly supported by the language but it can be either simulated [20] or implemented by an extension of the language syntax [1].

PDDL	Main features
1.0	STRIPS syntax
1.2	Object type hierarchy
2.1	Fluents and objective metrics
2.2	Derived predicates
3.0	Plan preferences
3.1	Object-fluents

Table 3.1: PDDL versions with introduced main features.

For a specific domain, a derived planning instance is obtained by the definition of two files: a domain file and a problem file. The domain file contains the problem structure that is common to the domain-specific problems; in particular it contains the predicates utilized to describe states and the definitions of the available actions. The problem file contains instead a specific problem instance, consisting by the available objects, the initial conditions and the goal state of the problem. Inside a problem instance, an object is identified by a unique name, and has a type specification (from PDDL version 1.2).

Actions have unique names and a list of typed parameters, pre-conditions and post-conditions. Actions pre-conditions are logical formulas that determines in which logical states of the system their utilization is possible, while post-conditions specifies a list of predicates that are set to true by their execution, and a list of predicates that are instead set to false.

Given a PDDL specification of a planning instance, the goal of a PDDL planner is to generate a sequence of actions that from the init conditions lead to the goal state. Furthermore, the generated plans should be optimal in the sense they should incur in the least cost, measured in number of performed actions or through a custom metric.

Specifically, a planning domain Ξ is specified by the tuple $\langle \mathcal{P}, \mathcal{A} \rangle$, where $\mathcal{P}$ is the set of possible truth predicates and $\mathcal{A}$ the set of of possible operators. A predicate $p \in \mathcal{P}$ is formed by a name and a list of arguments, i.e. $p = \langle name(p), arg_1(p), arg_2(p), ..., arg_n(p) \rangle$. Operators are formed by a name, a set of preconditions, and a set of effects in the form of deletions and additions, i.e. $a = \langle pre(a), eff^-(a), eff^+(a) \rangle$. An operator a is applicable to a state Σ only if the truth value of its preconditions evaluated on the state is positive; the state is then modified by removing from it the predicates in $eff^-(a)$ and adding the predicates in $eff^+(a)$.

For a specific planning domain $\Xi = \langle \mathcal{P}, \mathcal{A} \rangle$, a derived planning instance

Listing 3.1: PDDL planning domain and problem for a simple drinking scenario. A plan recognition algorithm could be able to detect that the observed agent is going to drink, and from which cup, after he grasped one of the three available ones.

```
(define (domain cups)
  (:requirements :strips :typing :existential-preconditions)
  (:types cup - object)
  (:predicates
    (drinked ?c - cup)
    (grasped ?c - cup))

  (:action grasp
    :parameters (?c - cup)
    :precondition (and (not (exists (?g - cup) (grasped ?g))))
    :effect (and (grasped ?c)))

  (:action drink
    :parameters (?c - cup)
    :precondition (and (grasped ?c))
    :effect (and (drinked ?c) (not (grasped ?c)))))


(define (problem cups-3-cups)
  (:domain cups)
  (:objects blue-cup yellow-cup red-cup)
  (:init)
  (:goal
    (or (drinked blue-cup)
        (drinked yellow-cup)
        (drinked red-cup))))
```

is obtained by specifying the tuple $\langle \Xi, I, G \rangle$. Where $I \subseteq \Sigma$ is the initial state, $G \subseteq \Sigma$ is the target goal. The goal of a planner is to create a plan $\pi \in \Pi$ that from I reaches G while incurring in the least cost, for example when the cost of a plan is measured by its length:

$$\pi = \underset{\pi \in \Pi}{\operatorname{argmin}} \ |\pi| + C(I, G, \pi) \tag{3.11}$$

where $C(.)$ is a constrain function which returns 0 if π is a valid plan starting from I and reaching G, ∞ otherwise. π, the found optimal plan, is a list of

operators $\{a_1, a_2, ..., a_n\}$ that allow to achieve G starting from I, or *Null* if a plan with bounded cost could not be found, i.e. $\forall \pi C(I, G, \pi) = \infty$.

From a planning instance $\langle \Xi, I, . \rangle$ and a sequence of observations $O = \{o_0, ..., o_n\} \subseteq A$, plan recognition is performed by computing, for every possible goal G, the cost of two optimal plans obtained from the planning instances $\langle \Xi', I, G' \rangle$ and $\langle \Xi'', I, G'' \rangle$, where:

- $\mathcal{A}' = \mathcal{A}$ with action effects modified as:
  $\forall a \in \mathcal{A}'$
  - $eff^+(a') = eff^+(a) \cup e_0$ if $a \in O$ and is the first of the observations (i.e. $n = 0$)
  - $eff^+(a') = eff^+(a) \cup e_{n-1} \rightarrow e_n$ if $a \in O$ and $n \geq 1$
  - $eff^+(a') = eff^+(a)$ otherwise.
- $G' = G \cup e_n$, where e_n is the effect predicate of the last action in O.
- $G'' = G$

The former planning instance achieves G while being constrained such that $O \subseteq \pi'$, the latter instead achieves G but is not constrained to generate plans that contain O. These two instances are used to compute the two costs $C(O, G)$ and $C(\emptyset, G)$ that are used to evaluate the degree of optimality $R(O, G)$ that the observations O have towards G; as it was described before in Section 3.1.

When using PDDL for plan recognition, computing $R(A, G)$ using a PDDL-based planner is rather straightforward, with a critical point being speed. PDDL planners are usually based on classical search algorithm (e.g. A*) that scale in a reasonable way with the problem size, with a worst case time complexity of $\mathcal{O}(2^n)$. Several improved search strategy have been proposed to partially cope to the scalability issues of PDDL, nevertheless, due to the fact that plan recognition requires $2|G|$ calls to the planner, a real-time speed is difficultly achieved. Plain PDDL based planning and plan recognition is suitable only for problems of small scale, also because due to the lacking of hierarchical decomposition of actions it is difficult to model and recognize higher level activities.

In Paper I the goal recognition techniques are based on PDDL as previously described in this section.

3.1.2 Plan recognition with Hierarchical Task Networks

Hierarchical Task Networks (HTNs) are hierarchical models which allow to define domains as an hierarchy of increasingly abstract tasks by recursively grouping sequences of atomic actions or tasks into macro tasks.

HTN problems are produced by the definition of a domain and a problem files. The domain file contains the available atomic and non-atomic tasks, together with a set of methods to decompose non-atomic tasks into sub-tasks. Similarly to PDDL, atomic tasks have concrete effects and preconditions. Instead, being abstract tasks a composition of sub-tasks, they inherit preconditions and effects in a bottom-up fashion.

In HTNs goals are specified as a target top level task to execute (rather than the desired goal state), or loosely as the post conditions that such task would produce. Since alternative sequence of tasks could be aggregated as forming the same top level task, the set of goal post conditions depend on which sub-tasks the planner decided to utilize to implement the goal task. The goal of the HTN planner is to find a sequence of task decompositions such that the goal task is decomposed to a sequence of atomic tasks, while incurring in the least plan cost (e.g. atomic plan length). The corresponding obtained plan is the sequence of atomic tasks that achieve the top level task.

Similarly as described in Section 3.1, goal and plan recognition can be performed by checking the degree of rationality that a gathered sequence of observations have with respect to the possible top level tasks in the HTN:

$$\forall t \in \mathcal{T} \;\; R(\hat{a}, t) = C(\hat{a}, t) - C(\emptyset, t) \tag{3.12}$$

With the goal being pursued by the agent selected by

$$\hat{t} = \underset{t \in \mathcal{T}}{\operatorname{argmin}} \;\; R(\hat{a}, t) \tag{3.13}$$

As described in [16], HTNs formalism allow for an alternative, efficient formulation of goal recognition, that requires a single call to the planner instead of $2|\mathcal{T}|$ as would happen if using Eq. 3.12. This is achieved by setting the tasks in $\mathcal{T}$ as candidate tasks for decomposition of an aggregate task $T_{\mathcal{T}}$. Then, with a single call, the planner is asked to plan for $T_{\mathcal{T}}$ while also complying with the gathered observations. Due to the fact that the planner prefers optimal plans, it will decompose $T_{\mathcal{T}}$ into the candidate goal task $t \in \mathcal{T}$ that is achieved most efficiently using the observations. This method utilizes a weaker rationality index

$$\forall t \in \mathcal{T} \;\; R(\hat{a}, t) = C(\hat{a}, t) \tag{3.14}$$

which measures how rational it is to pursue a goal *from the current state*, rather than from the beginning of the observations.

3.2 Plan recognition with deep networks

Another framework that allows goal and plan recognition is Deep Learning. Deep Learning got a lot of attention during past decade due to the ability for deep neural networks to learn from data, or create, complex mappings $P(Y|X)$. Usually, a neural network learns such mapping from a dataset of pairs (x_k, y_k), where x_k is an input for the network, y_k is the "correct answer" the network should learn to output when receiving x_k.

In Deep Learning, the huge amount of available data allows deep networks perform *representation learning*, that is they are able, thorough the learning process, to generate in the hidden layers features that are of an increasingly abstract level from the network's input, while holding their utility for the mapping the network is learning. For this reason, in deep learning it is rarely heard of feature or kernel engineering [5, 18, 3, 24], and problems are solved in a end-to-end fashion even when complex data (e.g. images, videos) is given.

To solve intent recognition using deep learning, we need datasets to train the neural networks. In particular, we need a plan library containing plans and associated goals. As also previously described, such dataset can be generated with the utilization of a planner. Algorithm 1 shows how a dynamic plan library can be generated [6].

Algorithm 1 Creation of a plan library of size N using a planner.

```
procedure MAKE-PLAN-LIBRARY(domain)
    while num_plans ≤ N do
        init ← SAMPLE-INIT(domain)
        goal ← SAMPLE-GOAL(domain)
        plan ← MAKE-PLAN(init, goal)
        yield (init, goal, plan)
    end while
end procedure
```

The dataset is composed of three pieces, each containing a part of the information required for plan recognition: the initial conditions, the goals and the plans. Thus, each record of the dataset has the form (i_k, g_k, a_k),

where a_k is a partial plan that from the initial conditions i_k is directed towards the goal g_k.

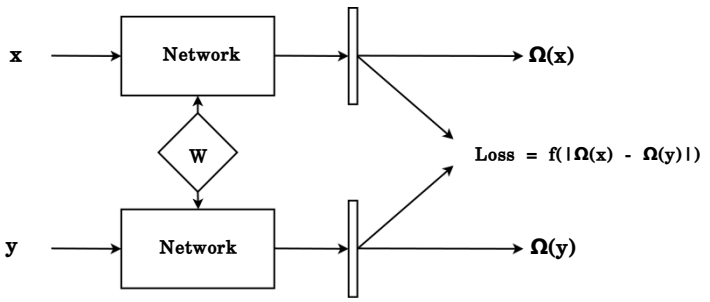

Figure 3.2: Schema of a Siamese Network.

The network we will utilize is a modified version of the Siamese Network [4, 17], that allows to perform both goal and plan recognition using the same architecture. This type of network shares the weights while processing pairs of input vectors. When providing pairs of the same type, the network should output vectors of high similarity, when pairs of different type are provided a low similarity instead. The similarity value can be obtained for example by evaluating the squared euclidean distance of the network's paired outputs:

$$out(x_0, x_1) = ||S(x_0) - S(x_1)|| \tag{3.15}$$

$$sim(x_0, x_1) = Sigmoid(out(x_0, x_1)) \tag{3.16}$$

where S is the output of the siamese network, $Sigmoid(.)$ is a fully connected layer with sigmoid activation. The siamese network loss function is obtained as:

$$L = -sim(x_{ak}, x_{ah}) + sim(x_{ak}, x_{bw}) \tag{3.17}$$

where x_{ah}, x_{ah} and x_{bw} are three different records of the dataset. a and b identify the class to which they belong to. Therefore, similarity between records of the same class is maximized, of different classes minimized.

We will use the siamese setting for evaluating pairs (i_k, g_k). Every pair corresponds to a different planning problem. The network will thus return points with high similarity for problems that are similar, with low similarity otherwise.

In addition, we specify a recurrent network $R(.)$ that will process, for every dataset record, (i_k, a_k). The output of the network has the the same

dimension of Eq. 3.15, therefore it is possible to compute similarites between plans and problem through Eq. 3.16. The loss function of the network is:

$$\begin{aligned} L = &-sim(R(i_k, a_k), S(i_k, g_k)) \\ &+sim(R(i_k, a_k), S(i_h, g_h)) \\ &+sim(S(i_k, g_k), S(i_h, g_h)) \end{aligned} \tag{3.18}$$

where (i_k, g_k, a_k) and (i_h, g_h, a_h) are records randomly sampled from the dataset. At inference time the network can be used to compute $P(G|A, I)$ (i.e. goal recognition) and $P(\Pi|A, I)$ (i.e. plan recognition) by the computation of similarities between the partial plans and goals, and between partial plans and the plan library. Given a set of goals G, a partial plan a and some initial confitions i_0, the probability distribution $P(G|a, i_0)P(a, i_0)$ is computed as:

$$\forall g \in G \;\; P(g|a, i_0) = \frac{sim(R(i_0, a), S(i_0, g))}{\sum_{g_n \in G} sim(R(i_0, a), S(i_0, g_n))} \tag{3.19}$$

To find the plan belonging to the plan library and best matching with the observed partial plan a we instead evaluate:

$$\forall \pi \in \Pi \;\; P(\pi|a, i_0) = \frac{sim(R(i_0, a), R(i_0, \pi))}{\sum_{\pi_n \in \Pi} sim(R(i_0, a), R(i_0, \pi_n))} \tag{3.20}$$

We tested siamese networks for goal recognition and compared with the baseline based on PDDL, yet unfortunately, no technical report is available for the moment. Our preliminary results showed that while accuracies between the two models are comparable in the case of no missing arguments in the plans' actions, PDDL based goal recognition holds a higher accuracy when we start removing actions arguments from the plans (actions thus become *partially instantiated*, see Paper I). Since our expectations were the opposite i.e. that siamese networks would be more robust in the case of missing data, further investigation that is left for future research is required.

3.3 Other recognition methods

Since intent recognition can be casted as plan or goal recognition or similarly formalized, a platora of methods present in the literature are utilizable. Some implementations of intent recognition may rely on classification:

having a set of labels for possible user intentions, observations can then be classified towards that label set. This approach is common in Natural Language Processing, where sentences are shallowly classified as the system actions desired by the user. In this case intentions don't directly belong to a task space in the form of a sequence of actions, but can rather be seen as user desires that the system should fulfill. For example, a uttered sentence "I am hungry" could be labeled as **user_desire**= *eat*. Since these classification methods don't utilize a model of the user or of the task we then categorize them as shallow methods.

Alternatively, several methods with similarities to plan recognition have been proposed in the literature. Such models comprehend, among many others, Partially Observable Markov Decision Process, Inverse Reinforcement learning of Spreading Activation Networks.

Analogously to plan recognition, this second group of methods relate the gathered observations to a task space that can be leveraged to compute plans by using a model of the agent-environment system. These methods allow to find, by using a corresponding rationality measure, what is the goal being pursued by the observed agent by the means of "minimum deviation". For example, in Inverse Reinforcement Learning the agent's plans and goals are found by "rolling out" its inferred policy, checking the states it reaches; or in Spreading Activation Networks the goals have likelihood proportional to the activation they receive from the active input nodes.

3.4 Grounding of observations

In intent recognition models, grounding the observations into the task space is a necessary step whenever they are not in a form that is directly utilizable. Observations must be grounded to representations that are understandable by the plan recognition component, before any inference can happen. For example, when using PDDL observations are grounded as operators, which can be utilized for plan recognition. In our formulation (see Eq. 3.2) grounding corresponds to specifying the model $P(A|O)P(O)$, which transforms the observation space O into the task space A. A set of obervations $\hat{o} \in O$ is grounded to a corresponding set of actions $\hat{a} \in A$ as:

$$\hat{a} = \sum_{A} P(a|\hat{o})P(\hat{o}) \quad \textbf{or} \quad \underset{A}{\operatorname{argmax}} P(a|\hat{o})P(\hat{o}) \tag{3.21}$$

Either models in Eq. 3.21 result in valid implementations for the grounding model.

As an example, raw data such as video, speech or other sensor modalities can be processed and integrated to obtain the representations belonging to the task space that will later be utilizable as sequence of recognized actions. For example, from raw video the online position of an observed person is first extracted and mapped to e.g. a grid map over which possible walking destinations can be evaluated.

Many possible algorithms exists for observations grounding, and we focus on hard-coded or neural learned methodologies. In hard-coded methodologies the observation and representation space are connected by a function that is manually created. In neural learning instead, this function is obtained by learning from data useful representations using neural networks.

For practical applications usually a mix of the two methods is often utilized. For example, using neural learning techniques complex data such as pictures or text may be reduced of dimensionality and clustered. The obtained binary representations can later be utilized as the discrete state that is needed by the logical part of the program, which usually is hand-crafted. This sort of combinations usually allow to insert into the system expert knowledge with the hand-crafted part, while dimensionality reduction allows an easier handling of the complex patterns that are inherently present in real-world data.

(Almost) purely neural methods are also possible as shown in [2], where the authors obtain a binary plan space in a end-to-end fashion by a combination of autoencoders and classical planners.

3.4.1 Speech

Speech interfaces allows a communication mechanism, through the utilization of a Spoken Dialogue Management System (SDMS) [22], that is easy to use by experimenters. Speech is a preferred mode of interaction with artificial systems such as in Human-Robot Interaction [22, 14, 23], where the SDMS translates utterances into semantic representations used to manage dialogues with the user, or that are forwarded to other system components. The input pipeline of a SDMS is constituted by an Automatic Speech Recognition (ASR) component, which transforms speech into text, and a Natural Language Understanding (NLU) component that classifies the textual utterances into semantic representations, such as speech acts [22]. The Dialogue Manager (DM) receives the parsing from the NLU and, also based on the interaction history, decides the system's output. The DM could eventually decide to respond to the user after an utterance, in this case, through a Nat-

ural Language Generation (NLG) component the semantic representations produced by the DM are converted back into a human-readable format, and eventually transformed to speech.

Spoken Dialogue Systems can be broadly categorized into two groups, open-ended and task-oriented. For task-oriented dialogues it is a common practice for SDMS to keep track of the goal the user has during dialogues. Traditionally, this goal is commonly expressed as some information the user wants to query from the system, or wants to insert into it. Goal recognition is then about inference of what the user is willing to query. A SDMS handling booking of flights or restaurants is a common example for this kind of dialogue managers [30]. In the context of SDMSs, the grounding process is thus achieved by defining a NLU component which generates the semantic structures that the DM can use to manage the dialogue.

When utterances should be mapped into tasks rather than queries, intent recognition requires to ground what the user is saying to elements belonging to the task the user is currently involved into. For example, in the context of an objects retrieval scenario, a user saying "Bring me the red box" could be translated to a semantic representation such as **bring(user, red-box)**. The user might want to open the box, or to put it in another location. By doing this, the intent recognition sub-system can carry inferences at the task level, rather than strictly at the dialogue level as in dialogue query systems.

Hence, uttered sentences are mapped at the task level into sub-tasks, that can allow to infer the macro task the user is belonging to, thus allowing intent recognition as plan recognition in the task space through e.g. HTNs. For the given example the inferred task will be the candidate task that contains the subtask **bring(user, red-box)** most optimally.

Suppose that in the object retrieval example there are two boxes, a green one and a red one, with the possible tasks being to load on a delivery truck either one of the two. Then, the utterance "Bring me the red box" can give evidence about the task the user is currently pursuing. In fact, the semantic translation of the utterance into **bring(user, red-box)** belongs better into a plan π_0 =**{bring(user, red-box), ..., load(red-box)}** rather than π_1 =**{bring(user, red-box), ..., load(green-box)}**.

Using the introduced grounding model (Eq. 3.21), utterances correspond to the observations O, while the tasks a user can possibly refer to are the tasks A. By shallowly parsing the sentence, Natural Language Processing tools such as Semantic Role Labeling can be used to associate verbs to tasks, and semantic roles to tasks arguments.

When using this method there is a clear distinction between the language

space of observations and the task space. In fact, the task space is arbitrarily linked on top of the utterances' semantic roles. In other formulations such as joint intention theory [19] dialogue and task can be entwined: dialogues are to be considered actions at the task level that are dedicated in manipulating intentions.

A problem formulation very similar to intent recognition is the one of generating robot commands from speech, where user utterances are to be translated into actions or full plans for a robot. A main difference is in how the utterance is interpreted: in the case of robot commands an utterance should result into an executable action for the robot, for intent recognition instead, the utterance specifies an action or an observation that belongs to an intention of the speaker. Notice that intent recognition subsumes direct robot commands. A recognized intention could contains actions that multiple agents are to perform. For example, a user and a robot could create a joint intention for washing dishes. If the robot is programmed to helping in acting out the inferred joint intention, then it should participate in it by taking charge of its assigned actions present in the joint intentions and execute them.

In Paper I we construct the PDDL operators used for plan recognition from the semantic roles contained in the uttered sentences.

3.4.2 Affordances

Affordances denote possibilities of action that the environment offer to the agents belonging to it through the means of their direct perception. Introduced by Gibson in 1966 in the field of Environmental Psychology, affordances have undergone a lot of discussion, both about its definition and applications [32, 13]. Informally, an affordance expresses a property of an entity such as being graspable, liftable, traversable, etc. that does not belong to the environment or the agent either, but rather to their intersection. It jointly expresses how the environment invites a certain behavior and the possibility of that behavior, that is; the environment affords a possible behavior to the agents that are able to perceive the corresponding affordance. A proper timely context should also be given: it was a heated period of investigation on whether internal representations where necessary for behavior to express itself. Gibson took the position that internal representations are ill posed, rather, behavior is representable by affordances that exists in the union of agents and environment, not in the agents alone!

Many interpretations have been proposed in past literature, and here

we provide the definition from Chemero [13], that is also compatible with the construction of computational models for affordance[2]. According to Chemero affordance are relations of the type:

$$\text{Affords–}\phi(\text{feature, behavior})$$

where ϕ is the affordance, feature is the relata that allows to detect the affordance perceptually, and behavior is the agent's skill that is associated to the relata. For example, a mug affords a grasping behavior to the agents able to perceive its graspable regions (such as the handle) to hold them. This can be expressed as:

$$\text{Affords–}\textit{graspability}(\text{mug, grasp})$$

Perceptions and behavior are inherently linked: the grasp behavior can unfold because of graspable regions, and regions of objects are graspable because a grasp behavior exists. When Gibsonians talk about affordance there is no separate definition for perception and behavior, but rather they conjoin in a unity expressed by an affordance that is directly perceived by the agents possessing the appropriate skill to perceive and hence enact the behavior. Behavior is perceived through the affordances it casts on its relata and viceversa (the relata is perceived through the behavior).

Perception can be casted on objects at different levels: meso affordances deal with whole objects (such as a door) while micro affordances with regions and micro behaviors. A subsumption hierarchy of affordances is expressible with relations of the type:

$$\text{Affords–}\theta(\text{Affords–}\phi_1, \text{ Affords–}\phi_2, \text{ ..., Affords–}\phi_n, \text{ behavior})$$

where θ subsumes $\phi_1, \phi_2, ..., \phi_n$ by the means of a particular behavior.

In computer science computational models at the meso level usually represent affordances as actor-behavior-object-effect relations. For example, a cup affords the action grasp or drink, a door the actions open or close. Different actor have access to different affordances: a baby cannot climb the stairs, but can grasp a toy.

Relations of the provided form allow to explore equivalences of actors, behaviors, objects and effects. Two affordances are said to be equivalent if their perceived effect is the same. For example, both a street and a river

[2]In this thesis we attempt to give an appropriate Gibsonian meaning. Discussions on how to interpret and bring the concept of affordance into computational models showed to be quite heated between research groups, and we invite the reader to refer to additional interpretations to have a more complete understanding of the concept.

are *traversable*. For a particular actor, ojects-behavior pairs are equivalent if they provide the same end result i.e. walk-street and swim-river are equivalent when the agent evaluates the traversability affordance. Different cups offer similar affordances, as the macro effects remain the same for the different actions the two mugs afford (drinking, grasping, etc).

A possible way to ground intentions is to use the affordances space. With this perspective, the partial plans taken by the agents correspond to trajectories of affordances activation. Plan recognition algorithms can thus be designed to keep track of the affordances activations, providing predictions about future activations.

Nevertheless, a domain's affordances must first either be gathered from the environment or specified; in Paper II we picked the former option and defined an algorithm that by leveraging natural language processing techniques gathers and generates possible names of actions associated to objects, with the goal of then linking this generation process to actual physical objects and actions, hence supporting intent recognition though eg. computer vision.

Chapter 4

Contributions

Given the relevant background present in Chapters 2–3, the contributions contained in this licentiate are toward the development of computational models for intention recognition by combining task planning and speech. Our main field of application is Robotics and Human-Robot Interaction.

4.1 Paper I

In the first contribution with title **Intent Recognition from Speech and Plan Recognition** we propose a method to combine natural language processing techniques with classical planning, to allow for inference of a speaker's intention from its utterances. The utterances are shallowly parsed using Semantic Role Labeling, to be then grounded into arbitrary predefined planning domains as commitments of the speaker. Once grounded, plan recognition algorithms make possible to infer the intention that the speaker is most likely referring to, in the form of an inferred goal and plan.

We extend the plan recognition algorithm proposed by [25] by explicitly allowing also partially instantiated actions to be present in the observations, rather than only fully instantiated traces. This allows for the speaker to instantiate observations through speech without having to provide values for all of the potential slots, that is a relevant point when developing speech interfaces for robotics.

We tested our modified algorithm on different planning domains utilizing only partially instantiated actions as observations. Then, we made a real

experiment with a Pepper robot to test its correct functioning in Human-Robot Interaction scenarios.

This contribution has been accepted at the *18th International Conference on Practical Applications of Agents and Multi-Agent Systems* [1].

My contribution

My contribution was to define, code and test the proposed methodology. I largely participated in writing the paper. I will participate at the PAAMS 2020 conference to present the contents of the paper.

4.2 Paper II

The second contribution **Unsupervised Object Affordances from Text Corpora**, is toward the construction of an intent recognition model grounded in the space of affordances. Affordances (see Section 3.4.2) express the link that exists between perceptions and actions, and our hypothesis is that they're are suitable to participate in the inference of intentions since they describe what is actionable by an agent and how. Nevertheless, before being able to be utilized, affordances must be either gathered from data or priorly specified.

This contribution is about the definition of a method to mine object-action pairs from textual corpora, to then create and test a generative model for candidate actions using a Conditional Variational Autoencoder (CVAE). By encoding the object-action pairs using word embeddings we first qualitatively test if and to which degree word embeddings contain information about affordance. We further show how on unseen objects the VAE outperforms a K–Nearest Neighbors baseline classifier.

Our idea was to concatenate the generative model to an object detection algorithm. Having the object in the current scene being recognized, we can then sample the CVAE to find possible actions to perform on the objects.

This contribution has been accepted at the *22nd Nordic Conference on Computational Linguistics* [2].

[1] www.paams.net
[2] www.nodalida2019.org

My contribution

My contribution was to define, code and test the proposed methodology. I also largely contributed in writing the paper. I participated at the NoDaLiDa 2019 conference to present the contents of the paper.

4.3 Paper III

Our third contribution **Traveling Drinksman** is about the creation of a service robot serving drinks to persons such as guests at a care house. Developed during my SOCRATES [3] secondment at Fraunhofer IPA, it is a piece of practical research which does not focus directly on intent recognition, but rather contains and describes a classical robot architecture with detection of humans, planning over tables and serving procedures.

We extended previous work by introducing two main components, a human detection module which allows to detect the room's tables that are occupied (i.e. with a human being seated nearby), and a global planner, that allows to plan most efficiently to serve all of the tables detected as occupied.

We were able to test the robot in real environments such as at a care-house in Stuttgart, as well as at the Fraunhofer IPA laboratory. Initial results proved to be promising, furthermore, the robot still has room to increase its functionality for example by adding social behavior such as chit-chatting.

This contribution has been accepted at the *52nd International Symposium on Robotics* [4].

My contribution

My contribution was to implement the high-level planning part of the system. Furthermore, I participated in the experiments at the care-house. I also largely participated in writing the paper. I will participate at the ISR 2020 conference to present the contents of the paper.

[3] The SOCRATES project is part of the European Union's Horizon 2020 research and innovation program. Please refer to *www.socrates-project.eu* for additional info.

[4] www.isr-robotics.org

4.4 Paper IV

Our fourth proposed contribution is an extended abstract with title **Mediating Joint Intentions with a Dialogue Management System**, where we propose a conversational mechanism to create joint intention with an artificial agent. Similarly as described in Chapter 5 joint intentions are defined as a shared plan and a goal. At every turn, the participants jointly contributes to the construction of the joint intention by either specifying a partially instantiated action the plan should contain, or by refusing a previously inserted actions. We further assume the human and robot participants share the same task model.

Since a dialogue manager based on a Finite State Machine might be too rigid in managing conversational interactions, we opted to utilize a Reinforcement Learning agent to learn the best conversational strategy. Our hypothesis is that this would allow for the artificial agent to learn a policy that is adapted over the interaction scheme of the human participant. However, since reinforcement learning requires hundreds or thousands of interaction to converge to a stable policy, the proposed system can difficultly be trained on real interactions. Therefore, we model the human participant as a fixed strategy that attempts to instantiate its goal intention with the least number of interactions, always being open to more efficient intentions proposed by the artificial agent.

This contribution has been accepted at the *First International Workshop on New Foundations for Human-Centered AI (ECAI-2020)*[5].

My contribution

My contribution was in jointly define, code and test the proposed methodology. I largely participated in writing the paper.

[5] https://nehuai2020.aass.oru.se/

Chapter 5

Future work

Intentionality still remains a crucial topic to be fully addressed in robotics. Taking into account also the discussions in the literature, in order to fulfill the HRI interaction requirements it seems that robots should be required to provide an *intentional interface*, rather than a physical one. For example, if taking BDI as blueprint for robotic agents, this interface could allow humans and other agents to directly perceive and hence interact with all of the three elements of the architecture. In other words, BDI agents could make available the affordances to directly interact with their beliefs, desires, etc. Also the robot itself could be programmed to directly perceive and act through this interface, whenever action on its own intentional state is required.

In the context of the mediation of a joint intention, what could be directly perceivable is the intention and what the interface provides is direct perception of how the intention can be mediated. It follows that the agents actions are not physical and instead belong to the intentional level. Nevertheless, this actions will indeed have a necessary physical manifestation. In this intentional context physicality is thus to be seen not as the action itself, but rather as only its manifestation.

This topic is not new, and how to infer mental states from physical interactions is what Theory of Mind is about. In fact, we could argue that what Theory of Mind strives for is direct perception of the mental, that can happen only through its physical manifestation. Parallelly, an effort to construct the operators of this interface is already attempted by Joint Intention Theory.

How to create such an interface which operates with mental concepts

(such as intention, beliefs and desires), but grounded to physical entities such as buttons, lights or physical movements? This might be what Theory of Mind applied to robotics could produce.

Given this general direction for future work, in the following sections we propose three topics as possible directions for this research, which instead focus on smaller contributions.

5.1 Deep models for goal recognition

As described in Section 3.2 deep networks can be employed for recognizing intentions. Our hypothesis is that similarly as in other fields such as in Computer Vision, they would provide similar advantages also for goal and plan recognition, namely, robustness to noisy observations and generalization capabilities. Nevertheless, while these properties are exhibited for very complex and rich data such as images, surprisingly they're difficult to reach for binary data such as PDDL plans. In fact, also our initial unpublished results show that classical planning methods still achieve a better overall performance.

Goal recognition using deep network is not new, but is found only as a classification task in the literature. Our proposed method would instead rely on siamese networks, which as well as allowing goal recognition, provide as end result of the training process an embedding space of plans.

As well as improving the goal recognition accuracies in the case of noisy observations (by using for example specifically tailored networks), future research on this topic could include how to engineer the siamese embedding spaces such that they exhibit relevant properties, such as embedding analogies, which are present in other embedding spaces like word embedding spaces.

5.2 Joint Intentions

We define joint intentions [29] as a direct extension to the definition of intention that we provided in Section 2. Joint intentions are intentions shared by multiple characters of the system, rather than belonging to single agents, and comprehend goals and plans that multiple agents agreed upon. Joint intentions are also sometimes referred to as "we" intentions, to better indicate that they belong to a "we" agent that multiple agents may form when cooperating.

Similarly as in Section 3, we define a joint intention as a plan $\hat{\pi}_{ab}$ together with a goal $\hat{g}_{ab}$. ab indicates that $\hat{\pi}, \hat{g}$ are relative to the joint intention that agents a and b share. Let us specify Π_a and Π_b as all the possible intentions agent a and b can have. In cooperative scenarios we can then argue that $\Pi_{ab} \supseteq \Pi_a \cup \Pi_b$. For example, in the box retrieval example a box might be too heavy for a single agent to lift, hence the action **load(heavy-box)** cannot belong to either Π_a or Π_b when the agents act alone, nevertheless, when a and b cooperate the same action can instead belong to their "we" agent, thus becoming a joint action. Of course, the joint agent implies a cooperative commitment by the two agents.

Since joint intentions belong to multiple agents, usually they require to be built and agreed upon. In Paper IV, we define a simple dialogue mechanism that two agents can use to mediate Π_{ab}. The algorithm leverage planning to structure the intention, and a Natural Language Understanding component based on Semantic Role Labeling to instantiate commitments towards Π_{ab}.

5.3 Model Reconciliation

When a human is asked to recognize the intention of a robot performing a task, it may sometimes be impossible for him to understand why it chose a particular sequence of actions rather than a more efficient alternative. The robot might indeed already be performing in the most optimal way, with being the human who cannot perceive this optimality. This a a case of model discrepancy, where the robot's and human's model of the task are different in a way such that the robot's actions are not explainable from the human perspective. It is in these cases that a model reconciliation might be necessary, where the model differences of the two models are inferred and explained by the robot, such that its actions become optimal also in the human's eyes. In recent research model reconciliation have been proposed to reconcile planning domains of human and robot that are relative to the same task, such as in [12] where the robot explains its plans in terms of differences between its and the human's task model.

We can think about applying model reconciliation techniques to plan recognition in two main ways: firstly, we can remove the assumption that agent and observer have the same task model—by for example using a second order Theory of Mind. In this scenario it would be required for the agent to produce explicable plans i.e. plans that are perceived as rational in the observer's task model, and hence, recognizable. Secondly, we can think about a dialogue manager that handles model reconciliations whenever the

observer cannot reliably infer the agent's plan.

Bibliography

[1] Armano, G., Cherchi, G., and Vargiu, E. An extension to pddl for hierarchical planning. In *Workshop on PDDL (ICAPS'03)* (2003), pp. 1–6.

[2] Asai, M., and Fukunaga, A. Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary. In *Thirty-Second AAAI Conference on Artificial Intelligence* (2018).

[3] Belkin, M., Ma, S., and Mandal, S. To understand deep learning we need to understand kernel learning. *arXiv preprint arXiv:1802.01396* (2018).

[4] Benajiba, Y., Sun, J., Zhang, Y., Jiang, L., Weng, Z., and Biran, O. Siamese networks for semantic pattern similarity. In *2019 IEEE 13th International Conference on Semantic Computing (ICSC)* (2019), IEEE, pp. 191–194.

[5] Bengio, Y., LeCun, Y., et al. Scaling learning algorithms towards ai. *Large-scale kernel machines 34*, 5 (2007), 1–41.

[6] Blaylock, N., and Allen, J. Generating artificial corpora for plan recognition. In *International Conference on User Modeling* (2005), Springer, pp. 179–188.

[7] Bonchek-Dokow, E., and Kaminka, G. A. Towards computational models of intention detection and intention prediction. *Cognitive Systems Research 28* (2014), 44–79.

[8] Bratman, M. E. Intention and personal policies. *Philosophical perspectives 3* (1989), 443–469.

[9] Byom, L. J., and Mutlu, B. Theory of mind: Mechanisms, methods, and new directions. *Frontiers in human neuroscience 7* (2013), 413.

[10] Cacace, J., Finzi, A., and Lippiello, V. Implicit robot selection for human multi-robot interaction in search and rescue missions. In *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)* (2016), IEEE, pp. 803–808.

[11] Carberry, S. Techniques for plan recognition. *User Modeling and User-Adapted Interaction 11*, 1-2 (2001), 31–48.

[12] Chakraborti, T., Sreedharan, S., Zhang, Y., and Kambhampati, S. Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. *arXiv preprint arXiv:1701.08317* (2017).

[13] Chemero, A., and Turvey, M. T. Gibsonian affordances for roboticists. *Adaptive Behavior 15*, 4 (2007), 473–480.

[14] Doshi, F., and Roy, N. Spoken language interaction with model uncertainty: an adaptive human–robot interaction system. *Connection Science 20*, 4 (2008), 299–318.

[15] Hart, J. W., Sheikholeslami, S., Pan, M., Chan, W. P., and Croft, E. A. Predictions of human task performance and handover trajectories for human-robot interaction. In *Proceedings of the HRI Workshop on Human-Robot Teaming (HRI'15)* (2015).

[16] Höller, D., Behnke, G., Bercher, P., and Biundo, S. Plan and goal recognition as htn planning. In *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)* (2018), IEEE, pp. 466–473.

[17] Koch, G., Zemel, R., and Salakhutdinov, R. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop* (2015), vol. 2, Lille.

[18] Koitka, S., and Friedrich, C. M. Traditional feature engineering and deep learning approaches at medical classification task of imageclef 2016. In *CLEF (Working Notes)* (2016), pp. 304–317.

[19] Kumar, S., and Cohen, P. R. Staple: An agent programming language based on the joint intention theory. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3* (2004), pp. 1390–1391.

[20] Mastrogiovanni, F., Sgorbissa, A., and Zaccaria, R. A system for hierarchical planning in service mobile robotics. In *Proc. of the Eighth Conf. of Int. Auton. Systems (IAS-8), The Netherlands* (2004).

[21] McDermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., Weld, D., and Wilkins, D. Pddl—the planning domain definition language, 1998.

[22] McTear, M. F., Callejas, Z., and Griol, D. *The conversational interface*, vol. 6. Springer, 2016.

[23] Milhorat, P., Lala, D., Inoue, K., Zhao, T., Ishida, M., Takanashi, K., Nakamura, S., and Kawahara, T. A conversational dialogue manager for the humanoid robot erica. In *Advanced Social Interaction with Agents*. Springer, 2019, pp. 119–131.

[24] Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[25] Ramírez, M., and Geffner, H. Probabilistic plan recognition using off-the-shelf classical planners. In *Twenty-Fourth AAAI Conference on Artificial Intelligence* (2010).

[26] Smith, M. C. Cognizing the behavior stream: The recognition of intentional action. *Child Development* (1978), 736–743.

[27] Sobel, D. M. Children's knowledge of the relation between intentional action and pretending. *Cognitive Development 22*, 1 (2007), 130–141.

[28] Sukthankar, G., Geib, C., Bui, H. H., Pynadath, D., and Goldman, R. P. *Plan, activity, and intent recognition: Theory and practice*. Newnes, 2014.

[29] Tomasello, M., Carpenter, M., Call, J., Behne, T., and Moll, H. Understanding and sharing intentions: The origins of cultural cognition. *Behavioral and brain sciences 28*, 5 (2005), 675–691.

[30] Wei, Z., Liu, Q., Peng, B., Tou, H., Chen, T., Huang, X.-J., Wong, K.-F., and Dai, X. Task-oriented dialogue system for automatic diagnosis. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (2018), pp. 201–207.

[31] Yazdani, F., Kazhoyan, G., Bozcuoğlu, A. K., Haidu, A., Bálint-Benczédi, F., Bessler, D., Pomarlan, M., and Beetz, M. Cognition-enabled framework for mixed human-robot rescue teams. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018), IEEE, pp. 1421–1428.

[32] Zech, P., Haller, S., Lakani, S. R., Ridge, B., Ugur, E., and Piater, J. Computational models of affordance in robotics: a taxonomy and systematic classification. *Adaptive Behavior 25*, 5 (2017), 235–271.

[33] Zhang, X., Yao, L., Huang, C., Sheng, Q. Z., and Wang, X. Intent recognition in smart living through deep recurrent neural networks. In *International Conference on Neural Information Processing* (2017), Springer, pp. 748–758.

Papers

Paper I	*Michele Persiani, Thomas Hellström.* **Intent Recognition From Speech and Plan Recognition**. *Proceedings of the 18th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS), pp. 212-223, 2020.*
Paper II	*Michele Persiani, Thomas Hellström.* **Unsupervised Inference of Object Affordance from Text Corpora**. *Proceedings of the 22nd Nordic Conference on Computational Linguistics (NoDaLiDa), pp. 115-120, 2019.*
Paper III	*Michele Persiani, Çağatay Odabaşı, Florenz Graf, Mohit Kalra, Thomas Hellström, Birgit Graf.* **Traveling Drinksman—A Mobile Service Robot for People in Care-Homes**. *Accepted at the 52nd International Symposium of Robotics (ISR), 2020.*
Paper IV	*Michele Persiani, Maitreyee Tewari.* **Extended abstract: Mediating Joint Intentions with a Dialogue Management System**. *Accepted at 1st New Foundations for Human-Centered AI Workshop (NeHuAI), 2020.*

This work has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 721619 for the SOCRATES project.

Paper I

Intent Recognition From Speech and Plan Recognition

Michele Persiani, Thomas Hellström

Proceedings of the 18th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS), pp. 212-223, 2020.

Intent Recognition from Speech and Plan Recognition

Michele Persiani( ) and Thomas Hellström

Umeå University, Umeå, Sweden
{michelep,thomash}@cs.umu.se

Abstract. In multi-agent systems, the ability to infer intentions allows artificial agents to act proactively and with partial information. In this paper we propose an algorithm to infer a speakers intentions with natural language analysis combined with plan recognition. We define a Natural Language Understanding component to classify semantic roles from sentences into partially instantiated actions, that are interpreted as the intention of the speaker. These actions are grounded to arbitrary, hand-defined task domains. Intent recognition with partial actions is statistically evaluated with several planning domains. We then define a Human-Robot Interaction setting where both utterance classification and plan recognition are tested using a Pepper robot. We further address the issue of missing parameters in declared intentions and robot commands by leveraging the *Principle of Rational Action*, which is embedded in the plan recognition phase.

Keywords: Intent recognition · Plan recognition · Natural Language Understanding · Semantic Role Labeling · Algorithms

1 Introduction

Intent recognition has been recognized as a crucial task in past and recent research in cybernetic systems [7,14,15], especially when humans are teaming along with artificial agents [5]. The ability to predict other agents' future goals and plans allows for proactive decisions, and relates to several system requirements, such as the need of an enhanced collaboration mechanism in human-machine interactions, the need for adversarial technology in competitive scenarios, ambient intelligence, or predictive security systems [5,15]. In this paper we focus on intent recognition for robotics, in scenarios where a person and a robot are present, yet the results have a broader applicability.

In robotics, the ability to predict users enables proactive behavior, ultimately giving the robots the ability to understand and coordinate actions with their users, even when only partial information is given [5,19]. In this paper we propose a method to infer user intent from speech, which is often a preferred mode of interaction in human-robot interaction [16]. Firstly, a series of utterances by the user are classified into partially instantiated PDDL [10] actions by using

Y. Demazeau et al. (Eds.): PAAMS 2020, LNAI 12092, pp. 212–223, 2020.
https://doi.org/10.1007/978-3-030-49778-1_17

Semantic Role Labeling [8]. The actions are then grounded into PDDL planning instances, and the user's intent is inferred using a plan recognition algorithm constrained to consider only plans containing the classified actions. The proposed method allows for discovery of intents beyond the scope of single sentences (achieved through, for example, a shallow classification of a sentence), by computing intents contextually to the task domain, in the form of a goal and a plan. Being able to reason on goals and plans using also context variables is necessary when attempting to describe or infer an agent behavior [5,14].

The rest of the paper is organized as follows. In Sect. 2 we introduce give background and related work for our proposed algorithm. Section 3 describes the intent recognition algorithm, followed by Sect. 4 in which we evaluate an implementation of the algorithm, both statistically by testing it in different planning domains, and experimentally using a Pepper robot. Finally, in Sect. 5 we give some conclusive remarks.

2 Background

In robotics, intent recognition can be performed using several modalities, such as video [9], gestures [12], eyeball movements, affect information in speech [4], and speech. When inferring intentions, raw input must be first transformed into data structures that are suitable for inference, such as action frames [2,18]. We refer to this as the process of grounding to the task domain. After grounding, various inference tools can be applied. For example, in [3], utterances are processed by mapping semantic roles into ad-hoc action frames using machine learning techniques. Semantic frames, such as the ones described in FrameNet [1], can be transformed to robot actions using sets of lexical units [18]. These units connect grammatical relations found in sentences to the different frame elements. With this approach, all core arguments must be present for the frames to be utilizable.

Pre-trained language models can also be utilized when inferring intentions from speech. Chen et al. [6] map semantic frames to robot action frames by using a language model trained on semantic roles, showing how large language models can be used to obtain the likelihood for the frames arguments. Their proposed *Language-Model-based Commonsense Reasoning* (LMCR) assigns a higher probability to the instruction *"Pour the water in the glass."* than to *"Pour the water in the plate."*. Thus, when the planning component is searching for an object to pour water into, it will prefer a glass rather then other objects. The LMCR is used to rank candidates for complete action frames by testing the different combinations of the available objects.

Inferred actions typically must have all arguments specified before they can be part of an executable plan. However, we can usually not expect all parameters to be fully specified in user utterances. In this regard our approach stands in contrast with other solutions (e.g. [6,17]) where the possible combinations of objects are exhausted or searched to retain only the most likely combination as candidate arguments. We instead allow for missing arguments to be present in the action frames, leveraging then the planner to infer them as the arguments that

would allow the whole inferred intention to be the least costly. Intentions are thus infused at parameter level with the *principle of rational action* i.e. intentional agents prefer optimal plans when evaluating different alternatives [19].

3 Method

We formally define an agent's intention as a goal $\hat{g}$ together with an action plan $\hat{\pi}$ the agent is committed to while pursuing $\hat{g}$ [19]. The sequence of actions $\hat{\pi}$ can either be a complete plan achieving $\hat{g}$ or a partial plan directed towards it. Intent recognition thus becomes the task of inferring $\hat{g}$ and $\hat{\pi}$ from a set of observations $o \in O$:

$$\hat{g}, \hat{\pi} = \underset{g \in G, \pi \in \Pi}{\operatorname{argmax}} P(g, \pi | o), \tag{1}$$

where G and Π are the set of possible goals and the set of possible partial plans respectively, O is the set of possible sets of observations. $\hat{g}$ and $\hat{\pi}$ are the arguments that maximizes the likelyhood of the intent recognition model $P(G, \Pi | O)$.

We additionally introduce an explicit grounding model $P(A|O)$ that is used to map raw observations to the task space as grounded actions. Furthermore, we add the assumption that the inferred plan is independent of the observations given the set of grounded actions $a \in A$. The formulation of the intent recognition model becomes:

$$P(G, \Pi | O) = \sum_{A} P(G|\Pi) P(\Pi|A) P(A|O) P(O). \tag{2}$$

Hence, a partial plan for the agent is first inferred from the grounded observations. Then, the plan is used to infer the agent's goal. Note that if the plan inference always infers complete plans, no inference of the goals is needed. Assuming that the agent behaves rationally, the inferred plan is the optimal plan achieving $\hat{g}$, and that contains the set of grounded actions $a \in A$.

We designed a method to infer the user's intention by grounding the utterances to sets of actions defined in a PDDL domain [10]. Semantic role labeling is used to extract semantic frames from the utterances. Each frame is then classified into a partially instantiated PDDL action to form the set $a \in A$. Inferred actions are then used to infer the speaker's intent $\hat{g}, \hat{\pi}$ using plan recognition.

Missing parameters in classified PDDL actions are automatically inferred by the planner as the ones that would make the speaker's inferred plan $\hat{\pi}$ least costly. For example, if the user utters "Give me something to drink" without specifying which glass to use, plan recognition will select the one that is most convenient to reach. The following example illustrates the process in more detail.

Parsing the utterance "Give me something to drink" may yield the following semantic parsing:

```
(define (domain cups)                    (define (problem cups-3-cups)
    (:requirements                           (:domain cups)
        :strips :typing :equality)           (:objects
    (:types cup - object)                        blue-cup yellow-cup red-cup)
    (:predicates                         ;;tag e:blue-cup   bow:blue,cup
        (finish ?c - cup))               ;;tag e:yellow-cup bow:yellow,cup
;;tag   e:drink bow:drink                ;;tag e:red-cup    bow:red,cup
;;roles e:drink role:ARG2                    (:init )
    (:action drink                       ;;goal (finish blue-cup)
        :parameters (?c - cup)           ;;goal (finish yellow-cup)
        :precondition ()                 ;;goal (finish red-cup)
        :effect (finish ?c))             )
)
```

Fig. 1. Example of specification of a PDDL domain and problem instances. In green the annotations performed on the entities $e \in E$. The annotations *tag* and *roles* allows to map bag of words into entities, while every *goal* annotation specifies a possible goal for plan recognition. (Color figure online)

- *verb:* give, *patient:* something to drink, *recipient:* me
- *verb:* drink *patient:* something

Assuming that the PDDL domain description contains the actions

- **(give ?to - agent ?i - item)**
- **(drink ?a - agent ?what - beverage ?from - item)**

the utterance may be classified as the partially instantiated actions

$$a = \{(\textbf{give me } None), (\textbf{drink } None\ None\ None)\}, \tag{3}$$

with the semantic roles of type *verb* mapped to the action names, and semantic roles *me* mapped to the first argument of **give**. Suppose that G contains two possible user goals: to be served food or to be served a drink. Then, the inferred plan $\hat{\pi}$ will have as goal to drink, as it is the least costly goal achieved with a plan constrained to contain a. Furthermore, when using partially instantiated actions the planner will select as the parameters that were set as ***None*** the objects belonging to the planning instance that would make the plan least costly.

3.1 Utterance Classification

For a given PDDL domain and problem definition, we define Act as the set of unique action names, and Obj as the set of all unique objects names. $E \subseteq (Act \cup Obj)$ is the selected subset of entities that are usable to instantiate PDDL actions from semantic roles. In order to map the semantic roles to an action parameter list in the correct order, we specify for every action $a \in (Act \cap E)$ a mapping between semantic roles and parameter indices:

$$M : A \times roles \rightarrow index \cup None. \tag{4}$$

For example, we can define that for the action **drink ?c - cup**, in the simplified drinking domain shown in Fig. 1, the semantic role *instrument* is associated to the 1st parameter. The mapping M allows to map semantic roles to the

parameters of the annotated actions. M is manually created by annotating the PDDL action descriptions.

Additionally, for finding the correct entities mentioned in the utterance we classify the semantic roles into entities by using a bag of words classifier. The training set for the classifier is obtained by manually annotating the PDDL domain. Figure 1 shows how a drinking domain is potentially annotated. Table 1 is the corresponding obtained dataset. Additional data is generated by data augmentation techniques (see Sect. 3.1) to improve generalization and robustness of classification. The dataset resulting from the annotation process contains records for the entities $e \in E$ only.

Table 1. Every action or object in the set of entities E is annotated with a bag of words that are used together with the object type as input for the entity classifier. E, the classifier's target label set, contains the PDDL unique names of the annotated entities.

$X_0 =$ **Bag of words**	$X_1 =$ **Type**	$E =$ **Id**
blue, cup	cup	blue-cup
red, cup	cup	red-cup
yellow, cup	cup	yellow-cup
drink	action	drink

For every record in the dataset, every word in $x \in X_0$ is encoded into its corresponding word-vector. $x \in X_1$ and $e \in E$ are categorical features encoded using one-hot-vectors. The target classes for the classifier are the unique PDDL labels of the entities in E. The described dataset is used to train a softmax classifier $P_e(E|X_0, X_1)$ that is used to instantiate PDDL actions from semantic roles by the following algorithm:

$$\hat{a} = \underset{e \in E}{\text{argmax}} P_e(e|b_{\mathbf{verb}}, \mathbf{action})$$

$$\forall i, \hat{e_i} = \underset{e \in E}{\text{argmax}} P_e(e|\{w\}_i, type_i), e_i \neq \hat{a}, M(\hat{a}, type_i) \neq None. \tag{5}$$

This sequence of classifications results in an action identifier $\hat{a}$ and a list of associated parameters $\{\hat{e}\}$. For a given action, not all of its semantic roles present in $M(\hat{a}, .)$ might be mentioned in the utterance and the missing ones will appear as *None* in the partially instantiated action. Additionally, semantic roles for which $M(\hat{a}, .) = None$ are discarded.

Notice that SRL could return multiple parsing for a given sentence, one for every verb it contains. In this case we run Algorithm 5 for every different parsing. This also allows to have multiple action declarations in the same sentence, such as in the case of *I'll go to the supermarket and buy maccaroni*, where SRL would produce a parsing for the verbs *go* and *buy*.

Data Augmentation. Data augmentation refers to a synthetic increase of the training data in order to increase the size of the dataset and thus the generalization capabilities for the trained model. For every entry in the original dataset we create $N = 1000$ synthetic entries by replacing, in every new record, the words in X_0 with random synonyms found using WordNet. Additionally, for every bag of word, N random words are added. Thus, the description of every object is expanded to the neighboring regions in word vector space by synonyms, while the injected random words increase the robustness of classification [20].

Negative Action Class. As described above, Algorithm 5 will always attempt to match bag of words with entities belonging to the problem. This is not always desirable, especially for auxiliary verbs such as *am* in phrases like *I am repairing my skateboard*, where SRL might label *am* as a verb and Algorithm 5 would thus return the action with similiar name (e.g. **eat**), resulting in a spurious action for the subsequent computations. For this reason, we allow for semantic roles to be classified as *None*. To detect such cases, the classifier is modified to allow the detection of outliers in its hidden layer, by a combination of regularization and Radial Basis Functions (RBF). In the case an input is detected as an outlier, the corresponding computation of the PDDL action or parameter is not performed.

In order to detect outliers, during training the classifier's hidden layer is regularized such that $\mathbf{h} \sim N(\mathbf{0}, \mathbf{1})$, as this helps in giving the data points a silhouette suitable for RBF when evaluated at the hidden layer of the classifier.

After training the regularized classifier, for every target class $e_i \in E$ a centroid c_i (and associated variance σ_i) is computed by averaging the vectors $\mathbf{h}$ generated by the training set. For every c_i only the rows with $e = e_i$ are taken. A Gaussian RBF network is then created with activation

$$\mathbf{a} = e^{\frac{-\|\mathbf{h}-\mathbf{c}\|^2}{\sigma^2}}, \tag{6}$$

with $\|.\|^2$ being the euclidean distance. Using the above defined RBF network, a bag of word is detected as outlier if $\max \mathbf{a} < T$, with T being a threshold hyper-parameter of the model.

3.2 Intent Recognition Through Plan Recognition

We apply a method similar to [11] that explicitly allows for partially instantiated actions to be present in the set of observations O, rather than allowing only fully instantiated ones. As the set of observations O we use the trajectory of past actions together with the partially instantiated actions gathered from sentence classification $a \in A$. We treat past observations and uttered actions in different ways, therefore splitting the set O into two parts, O_p and O_f. O_p is constrained to appear in a given sequence, as past observations are gathered in a specific order. For the uttered (possibly) future actions O_f no order is enforced instead.

From an instance $P = (G, I, A)$ (G: goal, I: initial conditions, A: available actions), a sequence of observed past actions O_p, and a set of partially

instantiated future actions O_f, we obtain two modified planning instances $P' = (G', I, A')$ and $P'' = (G'', I, A')$ that are used to compute $C[G + O]$ and $C[G + \neg O]$ respectively, where:

- $A' = A$ with action effects modified as:
  $\forall a \in A'$
  - effects(a') = effects$(a) \cup p_a \rightarrow e_0$ if $a \in O_p$ and is the first of the list (i.e. $n = 0$)
  - effects(a') = effects$(a) \cup p_a \wedge e_{n-1} \rightarrow e_n$ if $a \in O_p$ and $n \geq 1$
  - effects(a') = effects$(a) \cup p_a \rightarrow f_a$ if $a \in O_f$
  - effects(a') = effects(a) otherwise.
  - $p_a = \wedge_i (x_{ai} = arg_{ai})$ if arg_{ai} is specified for action i
  - $p_f = \cup_i f_i$
- $G' = G + O = G \cup e_n \cup p_f$, where e_n is the effect predicate of the last action in O_p, and p_f the conjunction of all of the effect predicates of the actions in O_f.
- $G'' = G + \neg O = G \cup \neg e_n \cup \neg p_f$

Every classified action $\hat{a}$ coming from the Natural Language Understanding component is inserted into the set of future observations O_f. Due to how partially instantiated actions are treated inside P' and P'', these actions receives an additional effect of the type

$$\wedge_i \, (x_{\hat{a}i} = arg_{\hat{a}i}) \rightarrow f_{\hat{a}}, \tag{7}$$

with $f_{\hat{a}}$ entering the set of goal predicates when computing $C[G+O]$. In this way, when computing this cost, the planner will also attempt to satisfy the actions $\hat{a}$ with the generated plan. For $C[G + \neg O]$ instead, the planner will be asked to not take actions $\hat{a}$. Notice that Eq. 7 is applied only to the parameters that are being specified in the action $\hat{a}$, and for which a valid semantic role was classified.

To compute the probability distribution for the goals, and hence of the intents, we pass the cost difference through a softmax layer obtaining $P(G_i|O) = \gamma e^{-\theta \Delta C_i} P(G)$, being $\Delta C_i = C[G_i + \neg O] - C[G_i + O]$, γ the normalizing factor and θ an hyper-parameter of the model, $P(G)$ the prior probabilities of the goals.

4 Evaluation

The evaluation of our proposed system is divided into two parts. Firstly, the developed plan recognition algorithm is evaluated statistically on different planning instances of high complexity. Statistical evaluation is done to quantify how partially instantiated actions alone contribute in the recognition of the correct goal. Then, we implement speech recognition together with image recognition on a Pepper robot, and evaluate intent recognition in human-robot interaction trials.

4.1 Evaluation of Plan Recognition with Partially Instantiated Actions

We evaluate our modified plan recognition algorithm, using only partially instantiated actions as observations, on the following planning domains. Our goal is to show how partially instantiated actions scale (i.e. how many specifications the user should give) when inferring goals in complex domains.

Logistics. In this well-known domain, a fleet of trucks and airplanes has to deliver packages from starting locations to destination ones. There exists different roads or flight routes in which subsets of trucks or airplanes belong to, and trucks and airplanes can move only in between nodes belonging their corresponding route system. The domain has 10 goals, each of them requiring to deliver 2 packages randomly picked from a set of 10 packages. There are 6 possible actions: *load-truck, load-airplane, unload-truck, unload-airplane, drive-truck, fly-airplane*, each of them having 3 arguments.

Blocks World. In this domain there is a table and several blocks on it. Blocks can be stacked on top of each other with the help of a gripper. There are 5 possible goals each of them being a set of towers of blocks. Only one action is possible, *stack-from-to*, that has 3 arguments.

Hospital. In the hospital domain a nurse has to inject drugs to the patients admitted at the hospital. Several rooms are dedicated for the patients and are spread over 3 floors. A set of elevators allow the nurse to change floor. The drugs are all initially stored in a storage room, and every patient requires a specific mixture of drugs. In addition, time constraints determine at which hour of the day the patients should receive their injections. The domain has 12 goals, each of them being the treatment of 2 patients. Patients, rooms, drugs and hours are chosen randomly when the domain is generated. There are 5 possible actions: *take-medicine*, *wait-for-hour*, *inject-drug*, *move*, *take-elevator*, with a mean number of arguments of 3.2.

For the three domains, each trial is carried as follow: a random goal is selected and an optimal plan A for it is generated. With a parameters $\alpha \in [0, 1]$ we selected the percentage of actions in A to keep and use for O_f (always at minimum one action was kept), with another parameter $\beta \in [0, 1]$ we specified the percentage of parameters to keep for every action. Retained parameters and actions are randomly selected at every trial. Every goal was tested in equal measure, and for every possible combination of α and β 10 trials were averaged.

Statistical results (Fig. 2) show how both α and β are important in plan recognition. When no parameter is specified ($\beta = 0$) the recognition gives the lowest accuracy values independently of α. In blocks world, being only one action present, this results in random guess performance; in logistics, this performance is slightly above random guess. Given that at least a parameter is specified ($\beta \geq \frac{1}{3}$ in our proposed scenarios), α becomes the dominating factor for the recognition accuracy, as better shown in the right column of Fig. 2.

For practical scenarios, a relevant case is when only one action is specified together with few parameters (e.g. $\alpha = 0$, $\beta \geq \frac{1}{3}$). In this case in the obtained accuracy is in the 20–60% range. Thus, if we expect a limited amount

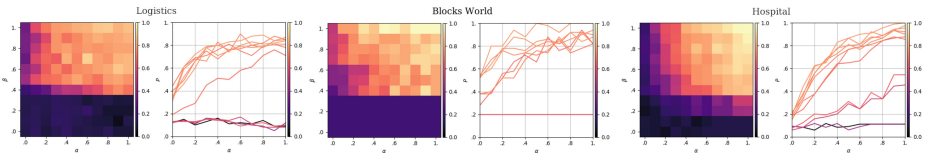

Fig. 2. Results of the statistical evaluation. The matrix on the left shows the tested combinations of values for α and β. Color, from black to white, indicates the obtained accuracy for every combination. On the right is plotted the accuracy in finding the correct goal using different values of α. Every different line correspond to a different value of β. (Color figure online)

of uttered commitments, the introduction of the set of ordered observations O_p is an important factor for achieving high accuracy. Nevertheless, notice that this is a pessimistic measure as in the benchmarks, actions and parameters are chosen randomly, while during real interactions we can expect the observed agent more likely to communicate informatively rather than randomly. Additionally, having the possibility of selecting the classifiable actions, we can ensure that only the actions that are pivotal for the plan recognition problem are expressible an utterances. No such constraint was present in the benchmarks.

4.2 Evaluation in an HRI Setting

In order to test intent recognition in an interaction with a robot, we implemented the described system in an HRI setting using a Pepper robot. In the proposed scenarios, an experimenter stands in front of Pepper and interacts with it using speech. Utterances are detected through the Google Speech API, and classified into PDDL actions using Algorithm 5. Additionally, based on the presence of different objects in the current visual scene, the truth value of selected predicates inside the planning instance is modified. Visual objects are detected using a classifier pre-trained on the YOLO dataset [13]. Figure 3 shows the full developed architecture.

Two different scenarios are evaluated: a *Groceries* scenario where inference on contextual elements is used to discriminate between the user intentions buying food or buying cigarettes. The second *Cups* scenario is created to verify how, given an utterance with partial specifications, missing parameters in the corresponding PDDL action are correctly inferred.

Groceries Setting. In this hypothetical setting the planning instance is programmed to detect whether the speaker in going to buy groceries or cigarettes. Through every trial, the user is asked to state what he is going to do. The possible choices are to buy from the grocery store, to eat food, or to smoke. The possible goals are to eat or to smoke.

Depending on the presence of food on the table in front of Pepper, the corresponding predicates expressing availability for that particular food are set inside the planning instance.

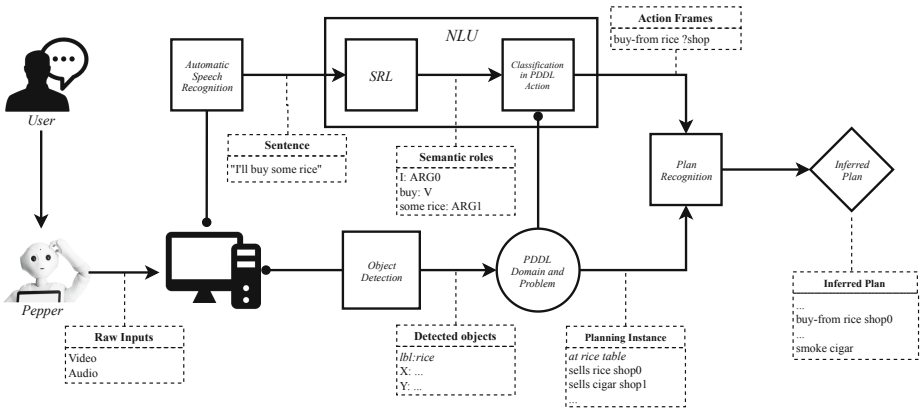

Fig. 3. Main architecture of the implemented system. Audio and video from the Pepper robot are streamed to a workstation where visual objects are identified and audio converted to text. Detected objects are used to modify the planning instance, while speech is classified into partially instantiated actions. The result is used to infer the speaker's intent through plan recognition.

The annotation of the PDDL domain and problem with semantic roles and bag of words is performed in a similar fashion as the one shown in Fig. 1. The expected outcomes of the trials are:

- If the user utters that he wants to go to the supermarket or buy food, the inferred goal depends on the predicate **(at rice fridge)**, which is set to true if a visual object of type *cup* or *bowl* is detected. In such case, the inferred goal is set to smoke, and otherwise to eat.
- If the user utters that he wants to cook or eat, the inferred goal is to eat, expressed by the predicate **(consumed rice)**.
- If the user utters that he wants to smoke, the inferred goal is to smoke, expressed by the predicate **(consumed cigar)**.

Cups Setting. In this setting the user can ask for a drink from three different cups on the table, each one with a different associated cost to reach. The only action that is accessible through speech is *drink*, with one optional parameter specifying which cup to use. There are three possible goals, achieved by the drink action using the different cups. The expected outcomes of the trials are:

- If the user says that he wants to drink, without specifying a cup, the goals have equal probabilities as no discriminating information is present. The inferred goal is returned as to drink from the blue cup.
- If the user specifies any cup for drinking, the inferred goal is to drink with the mentioned cup.

During the experiments the algorithm behaved as expected, and the robot inferred different intentions based on the perceived contextual variable.

A video showing the different experimental trials for both scenarios is available at https://youtu.be/33Dinfh7_0Y (please make sure the address is properly typed).

5 Conclusions

W proposed an algorithm to infer a speaker's intention from utterances and context. The proposed method is based on the classification of the utterances into PDDL actions, followed by a plan recognition algorithm using classical planning. Matching of parts of the utterance to actions and parameters is done using semantic role labeling. Recognized utterances are used to infer the partial plan and goal of the speaker, or to guide execution of actions when part of the information is missing. The proposed system allows to utilize utterances in a contextual way, and depending on the state of the planning instance they lead to different inferred intentions. In our HRI experiments the robot reacts to the user utterances by simply telling the goal it inferred. More complex type of reactions are also possible and are left for future research. The major benefit with our approach is that the intentions do not have to be hardcoded for combinations of a large number of contextual states, but is rather intelligently inferred by the robot in a way that scales both with number of possible intents and contextual variables.

We discuss the issue that when instantiating robot commands all of required parameters must be present in order for the commands to be executed. With the support of a planning domain, partially instantiated actions allow instead to take advantage of the principle of rational action, thus inferring missing parameters as the ones that would yield the most optimal intention. This method is in contrast with other approaches where the combinations of available objects are exhausted or searched in order to find the best match.

Evaluation showed how partially instantiated actions positively contribute to inference of the correct goal. For complex scenarios they yield a fair accuracy only when present in fairly large numbers. Additionally, the system was implemented in an HRI setting using a Pepper robot, and we verified its correct operation in several simplistic but relevant experiments.

Future research include incorporation of a dialogue manager to create/mediate intentions, of multiple agents in the inferred intentions, and collection of a structured knowledge-base for planning domains and annotations, possibly testing grounding algorithms that generalize over them.

Acknowledgments. This work has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 721619 for the SOCRATES project.

References

1. Baker, C.F., Fillmore, C.J., Lowe, J.B.: The berkeley framenet project. In: Proceedings of the 17th International Conference on Computational Linguistics-Volume 1, pp. 86–90. Association for Computational Linguistics (1998)

2. Bastianelli, E., Castellucci, G., Croce, D., Iocchi, L., Basili, R., Nardi, D.: Huric: a human robot interaction corpus. In: LREC, pp. 4519–4526 (2014)
3. Bensch, S., Jevtić, A., Hellström, T.: On interaction quality in human-robot interaction. In: International Conference on Agents and Artificial Intelligence (ICAART), pp. 182–189 (2017)
4. Breazeal, C., Aryananda, L.: Recognition of affective communicative intent in robot-directed speech. Auton. Robots **12**(1), 83–104 (2002). https://doi.org/10.1023/A:1013215010749
5. Chakraborti, T., Kambhampati, S., Scheutz, M., Zhang, Y.: Ai challenges in human-robot cognitive teaming. arXiv preprint arXiv:1707.04775 (2017)
6. Chen, H., Tan, H., Kuntz, A., Bansal, M., Alterovitz, R.: Enabling robots to understand incomplete natural language instructions using commonsense reasoning. CoRR (2019)
7. Demiris, Y.: Prediction of intent in robotics and multi-agent systems. Cogn. Process. **8**(3), 151–158 (2007). https://doi.org/10.1007/s10339-007-0168-9
8. He, L., Lee, K., Lewis, M., Zettlemoyer, L.: Deep semantic role labeling: what works and what's next. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 473–483 (2017)
9. Kelley, R., Browne, K., Wigand, L., Nicolescu, M., Hamilton, B., Nicolescu, M.: Deep networks for predicting human intent with respect to objects. In: 2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp. 171–172, March 2012
10. McDermott, D.: PDDL-the planning domain definition language (1998)
11. Ramírez, M., Geffner, H.: Probabilistic plan recognition using off-the-shelf classical planners. In: Twenty-Fourth AAAI Conference on Artificial Intelligence (2010)
12. Rani, P., Liu, C., Sarkar, N., Vanman, E.: An empirical study of machine learning techniques for affect recognition in human-robot interaction. Pattern Anal. Appl. **9**(1), 58–69 (2006)
13. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
14. Schaefer, K.E., Chen, J.Y., Wright, J., Aksaray, D., Roy, N.: Challenges with incorporating context into human-robot teaming. In: 2017 AAAI Spring Symposium Series (2017)
15. Sukthankar, G., Geib, C., Bui, H.H., Pynadath, D., Goldman, R.P.: Plan, Activity, and Intent Recognition: Theory and Practice. Newnes, London (2014)
16. Teixeira, A.: A critical analysis of speech-based interaction in healthcare robots: making a case for the increased use of speech in medical and assistive robots. In: Speech and Automata in Health Care, pp. 1–29 (2014)
17. Tellex, S., et al.: Understanding natural language commands for robotic navigation and mobile manipulation. In: Twenty-Fifth AAAI Conference on Artificial Intelligence (2011)
18. Thomas, B.J., Jenkins, O.C.: Roboframenet: verb-centric semantics for actions in robot middleware. In: 2012 IEEE International Conference on Robotics and Automation, pp. 4750–4755. IEEE (2012)
19. Tomasello, M., Carpenter, M., Call, J., Behne, T., Moll, H.: Understanding and sharing intentions: the origins of cultural cognition. Behav. Brain Sci. **28**(5), 675–691 (2005)
20. Wei, J.W., Zou, K.: Eda: Easy data augmentation techniques for boosting performance on text classification tasks. arXiv preprint arXiv:1901.11196 (2019)

Paper II

Unsupervised Inference of Object Affordance from Text Corpora

Michele Persiani, Thomas Hellström

Proceedings of the 22nd Nordic Conference on Computational Linguistics (NoDaLiDa), pp. 115-120, 2019.

Unsupervised Inference of Object Affordance from Text Corpora

Michele Persiani
Department of Computing Science
Umeå University
Umeå, Sweden
michelep@cs.umu.se

Thomas Hellström
Department of Computing Science
Umeå University
Umeå, Sweden
thomash@cs.umu.se

Abstract

Affordances denote actions that can be performed in the presence of different objects, or possibility of action in an environment. In robotic systems, affordances and actions may suffer from poor semantic generalization capabilities due to the high amount of required hand-crafted specifications. To alleviate this issue, we propose a method to mine for object-action pairs in free text corpora, successively training and evaluating different prediction models of affordance based on word embeddings.

Affordance; Natural Language Processing; Robotics; Intention Recognition; Conditional Variational Autoencoder;

1 Introduction

The term "affordance" was introduced by the American psychologist Gibson (Greeno, 1994) to describe what an animal can do in a given environment. It has since then been extensively utilized, interpreted, and re-defined (see (Çakmak Mehmet R. Doğar et al., 2007) for an overview) in fields such as robotics (Zech et al., 2017), human-computer-interaction (Schneider and Valacich, 2011) or human-robot-interaction (HRI) (E. Horton et al., 2012). Several interpretations for affordance exist in the literature, we use the term in a loose way to denote actions that can be performed with objects. As a simplified first approach we assume a one-to-many mapping G: *Objects* $\rightarrow$ *Affordances*. The object "door" may, for example, be used to perform the actions "open", "close", and "lock".

This paper presents how G may be learned from free-text corpora. The results show how it is possible to learn a generative model G that, given an object name, generates affordances according to a probability distribution that matches the used training data. Qualitatively results also indicate that the model manages to generalize, both to previously unseen objects and actions.

The paper is organized as follows. In Section II and III we give a brief literature review on affordances from different fields. The developed method is described in Section IV, and results from the evaluation are presented in Section V. The paper is finalized by conclusions in Section VI.

2 Affordances

When learned, the mapping G can be used in several ways in artificial systems, for example, by visually identifying objects in the environment or in the verbal dialogue with the user, suitable actions can be inferred by applying G to the observed objects. The objects and actions can then be used for shared planning or intent recognition (Bonchek-Dokow and Kaminka, 2014), thus allowing closer cooperations with the user.

For example, the mapping G may be used in a robot to decide how it should act within a given context that affords certain actions. In HRI, a service robot may for example suggest its user to read a book after it being visually detected or mentioned. Affordances may also be useful for object disambiguation. When a robot is told to "pick it up!", the robot only has to consider objects that are "pickable" in the current scene (E. Horton et al., 2012). Alternatively, affordances may be used to infer the human's intention, which may guide the robot's behavior (Bonchek-Dokow and Kaminka, 2014). If a user expresses will of talking to his children, a robot may infer that the user want to call them, and suggest making a phone call. Inference of affordances may also be used to design robots that are understandable by humans, since mutually perceived affordances may contribute to explaining a robot's behavior (Hellström and Bensch, 2018), and thereby increase interaction quality (Bensch et al., 2017).

Classical planning require knowledge about the actions that are possible in a certain situation, i.e. its afforded actions. For simple scenarios, it could suffice to enumerate all objects in the current scene, to later score their affordances and finally select the most promising to activate.

Affordances can be organized in a hierarchy, thus exposing relations or subsumptions between actions (Antanas et al., 2017; Zech et al., 2017). Assuming that a door affords the action *open*, it is clear that in order to be opened, several actions must be performed in a precise sequence (e.g. turn the handle, push the handle). Objects that offer the same grouped sequence of actions could then be represented as similar in a latent

space.

Antanas et al. (Antanas et al., 2017) relate affordances to the symbol grounding problem. In the attempt of grounding the object *door*, we could say it is an object affording *open, close,* etc.: it is grounded over those actions. Further stress is also put on describing affordances as relations between objects and qualities of objects. A pear can be cut with a knife because it's soft, while a hard surface could instead be just scraped. The blade of the knife affords cut only if used in conjunction with soft enough objects. This relational hypothesis is supported by neuroscience studies showing how motor cortices are activated faster if a tool is presented together with another contextual object, rather than alone (Borghi et al., 2012).

Depending on the desired level of abstraction, affordances can be represented on different levels (Zech et al., 2017). We broadly distinct two categories, namely symbolic and sub-symbolic. In symbolic form, affordances are expressed through symbols, and every symbol enjoys certain relations with other symbols. This usually gives rise to the possibility of having a knowledge-base, containing entities such as *affords(knife, cut, pear)*, and organizing them in a graph. Sub-symbolic encodings (such as through neural networks) are instead useful to obtain percepts (Persiani et al., 2018). By clustering the perceptual/procedural space, we obtain entities (the centroids) that may or may not be utilizable as symbols, depending on the nature of the input space and subsequent calculations.

Inference of affordances from images (Zech et al., 2017) is an example of sub-symbolic approach. This is related to object recognition/segmentation, and corresponds to associating afforded actions to different visual regions of the object. Recognized affordance regions can be used for object categorization (Dag et al., 2010). For example, in a kitchen environment objects having two graspable regions could be identified as pans or containers. This is especially useful for robotic manipulation tasks (Yamanobe et al., 2017): a planner for a gripper must have knowledge about the geometric shape of the parts that can actually be grasped.

Ruggeri and Di Caro (Ruggeri and Caro, 2013) propose methodologies on how to build ontologies of affordances, also linking them to mental models and language. If we think at the phrase "The squirrel climbs the tree", we can create a mental image for it, imaging how it reaches the top. If an elephant climbs the tree instead, surely some semantic mismatch will soon arise. The mental model doesn't fit because the tree doesn't afford climbing to the elephant. The opposite might instead apply for scenarios like *"Lifting a trunk"*.

3 Related work

Unsupervised extraction of object-action pairs from free text corpora has been a relevant point in recent Natural Language Processing (NLP) research. Differently from the other methods, corpora can be mined by different techniques with the goal of finding in an unsupervised manner relationships between objects, properties of objects and actions. Chao et al. (Chao et al., 2015) show how in NLP objects and actions can be connected through the introduction of a latent space. They argue that building such a space is equivalent to obtaining a co-occurrence table, referred to as the "affordance matrix". In their approach every object-action word pair is scored through a similarity measure in the latent space, and only the pairs over a certain threshold are retained as signaling the presence of affordance. The affordance matrix, together with other automatically extracted properties and relations (altogether referred to as commonsense knowledge), such as expected location for objects, can be then used to build PKS (Planning with Knowledge and Sensing (Petrick and Bacchus, 2002)) planners (Petrick and Bacchus, 2002; Kaiser et al., 2014).

In (Chen et al., 2019), the authors map semantic frames to robot action frames using semantic role labeling, showing how a language model can yield the likelihood of possible arguments. Their proposed *Language-Model-based Commonsense Reasoning* (LMCR) will give as more probable an instruction such as *"Pour the water in the glass."* rather than *"Pour the water in the plate."*. The LMCR is trained over semantic frames by using mined knowledge about semantic roles and can be used to rank robot action frames by testing the different combinations of the available objects. When searching for an object where to pour water, the LMCR is used to rank the available objects.

4 Method

We trained a generative model for the one-to-many mapping $G : Objects \rightarrow Affordances$ using pairs of the type <*object*, *action*>. These pairs were generated by *semantic role labeling* of sentences from a selected corpus. Objects and actions were represented by *wordvectors* throughout the process, as is illustrated in Fig. 1. The model allows to rank the different affordances for a given object name, as names of actions that can per performed on it. By employing a neural network model rather than a tabular model we investigate whether wordvectors encoding allows for the generalization the in mapping object-action.

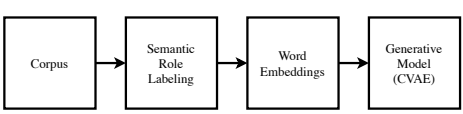

Figure 1: Steps taken to obtain the generative model.

4.1 Corpus

As data source we used the *Yahoo! Answers Manner Questions* (YAMC) dataset[1] containing 142,627 ques-

[1]Obtained at *https://webscope.sandbox.yahoo.com/catalog.php?datatype=l.* Accessed May 16, 2019.

tions and corresponding answers. The corpus is a distillation of all questions gathered from the platform *Yahoo! Answers* during the year 2007. It is a small subset of all questions, selected for their linguistic properties such as good quality measured in terms of vocabulary and length.

This specific corpus was selected due to the nature of its content. Our hypothesis is that being a collection of QA regarding daily living, the actions and objects being mentioned are more closely related to affordance than the ones in other corpora such as Wikipedia.

4.2 Semantic Role Labeling

In NLP, semantic roles denote the semantic functions that words have in a given phrase (Carreras and Márquez, 2004). For example, in the phrase "John looks in the mirror", the words "looks in" (denoted V) refer to the action being performed. "John" identifies the agent carrying out the action (denoted $A0$), and "the mirror" is the object (denoted $A1$) being target of the action.

Semantic role labeling (Gildea and Jurafsky, 2002) is the task of assigning semantic roles to words or groups of words in a sentence. A variety of tools exist for this task, with different conventions for the associated roles. As an example, for (Sutherland et al., 2015), the SEMAFOR parser (Das et al., 2010) was used to infer human intention in verbal commands to a robot. In the current paper we used the parser in SENNA (Collobert et al., 2011), which is a software tool distributed with a non-commercial license.

After parsing the corpus using SENNA, phrases with semantic roles $A1$ and V of size one were selected. Each action V was lemmatized into the basic infinitive form since we were not interested in discriminating temporal or other variants of the verbs.

Finally, all pairs ($A1$,V) that appeared at least seven times were used to create data samples <*object, action*>. This number was found to filter out spurious pairs. A fictional example illustrating possible generated sample pairs <*object, action*> is shown in Table 4.2.

Phrase	<**object, action**>
Add flour.	<flour, add>
Crack the egg.	<egg, crack>
Set the mixer on two steps.	<mixer, set>
Whip using the mixer.	<mixer, use>
Open the oven.	<oven, open>
Enjoy the cake.	<cake, enjoy>

Table 4.2 Examples of object-action pairs generated from phrases in a recipe.

Objects and actions are further filtered based on a *concreteness* value (Kaiser et al., 2014), that correspond to how close they are to being physical entities rather than abstract ones. To do so, for every sense of every object we navigate the WordNet entity hierarchy and retain that sense only if it is a child node of *physical entity*. Only objects with a ratio of physical senses above a certain threshold are kept. We apply the same procedure to actions but regarding them as physical if they are child of *move, change, create, make*.

4.3 Dataset

The words in each generated pair <*object, action*> were converted to wordvectors to provide numeric data to be used in the subsequent experiments. All data was divided into a training set comprising of 734,002 pairs, and a test set comprising 314,572 pairs. Special care was taken to include different objects in training and test data sets. This would allow us to test in a more aggressive way the generalization capabilities of the trained models. The data contained $N_O = 33{,}655$ distinct object names and $N_A = 11{,}923$ distinct action names.

4.4 Word Embeddings

Word embeddings (Collobert et al., 2011) model every word x as a dense vector W_x. Words that co-occur often in the corpus have similar associated vectors, and enjoy linear or non-linear properties reflecting semantic or syntactic relationships such as analogies(Drozd et al., 2016). $W_{king} - W_{man} \approx W_{queen} - W_{woman}$ (semantic analogy), or $W_{lift} - W_{lifted} \approx W_{drop} - W_{dropped}$ (syntactic analogy). Similarity of words is often measured though cosine distance of the vectors. For a review on analogy tests see (Finley et al., 2017).

GloVe (Pennington et al., 2014) and Word2Vec (Mikolov et al., 2013) are common approaches to create word embeddings. We trained Word2Vec over YAMC to get embeddings for words that were most specific for our dataset. The selected dimensionality for the wordvectors was 100.

4.5 Generative Model

We compare three different models in how good they are in predicting $P(A|O)$ provided the evidence in the data. A *Conditional Variational Autoencoder* (CVAE) (Doersch, 2016) trained on off-the-shelf GloVe embeddings with dimensionality 200, a CVAE trained on word2vec embeddings fitted on the YAMC dataset, a *K-NN* model.

4.5.1 Conditional Variational Autoencoder

A CVAE is a trainable generative model that learns a conditional probability distribution $P(A|O)$ while keeping a stochastic latent code in its hidden layers. They can be divided into two coupled layers: an encoder and a decoder. The encoder transforms the input distribution into a certain latent distribution $Q_\phi(z|A, O)$, while the decoder reconstructs the original vectors from its latent representation z together with the conditioning input o, with output distribution equal to $P_\varphi(A'|z, o)$.

The encoder's latent layer is regularized to be close to certain parametric prior $Q_\vartheta(z|O)$. The lower-bound

loss function for the CVAE is:

$$L_{CVAE} = \mathbb{E}[\log P_\varphi(A'|z,o)] - \lambda D_{KL}(Q_\phi(z|A,O)||q_\vartheta(z|O)) \quad (1)$$

The first term accounts for how good the autoencoder reconstructs the input given its latent representation. The second term regularizes the hidden latent space to be close to a certain posterior distribution. The factor λ balances how regularization is applied during learning. Starting from zero it is linearly grown up to one as the learning epochs advance. This technique addresses the *vanishing latent variable problem* and is referred to as KL annealing (Bowman et al., 2016).

φ, ϕ, ϑ denotes the three disjoint sets of parameters of the components that are simultaneously involved in learning. More specifically, they represent set of weights for the three neural network composing the CVAE. The CVAE was trained using the training set generated as described above, and was implemented using the Keras (Chollet et al., 2015) library for Python.

In order to search for a most direct relationship between objects and actions in wordvectors space, we keep the autoencoder with one hidden layer in both encoder and decoder. Nevertheless, nonlinearity of the output function of the hidden units proved necessary to yield a high accuracy. We set the dropout value for the hidden layers of the autoencoder to 0 (no features are dropped during the training phase), as this setting proved better performance in all of the experiments.

4.5.2 Nearest Neighbor

For a given input object o, the Nearest Neighbors model predicts $P(A|o)$ as $P(A|o')$, where o' is the closest object in training data. o' is found by cosine similarity of the wordvectors o and o'. $P(a'|o')$ is computed as $N(a', o')/N(o')$, where $N(.)$ is the counting of occurrences in training data.

Input	Output
door	open, pull, put, loosen, grab, clean, leave, get, slide, shut
egg	hatch, poach, implant, lay, crack, peel, spin, whip, float, cook
wine	pour, add, mix, dry, rinse, melt, soak, get, use, drink
book	read, get, write, purchase, find, use, sell, print, buy, try
cat	declaw, deter, bathe, bath, spay, pet, scare, feed, attack
money	loan, inherit, double, owe, withdraw, save, waste, cost, earn, donate
knife	scrape, cut, brush, chop, use, roll, pull, remove, slide, rub
body	trick, adapt, tone, adjust, recover, starve, cleanse, respond, flush, exercise

Table 4.5.2 Examples of actions generated by the CVAE. For every input object the 10 most probable outputs are sorted from high to low probability.

5 Evaluation

By sampling the model, we obtain names of possible actions A. As described above, the sampling follows the estimated conditional probabilities $P(A|O)$. Hence, actions with high probability are generated more frequently than actions with low probability. Since the CVAE outputs actions in numeric wordvector format, these actions are "rounded" to the closest action word appearing in the dictionary. This is equivalent to a *K-NN* classification with $K = 1$. A few examples of the most probable generated actions for CVAE are shown in Table 4.5.2.

Evaluation of generative models is in general seen as a difficult task (Theis et al., 2015; Hendrycks and Basart, 2017; Kumar et al., 2018), and one suggestion is that they should be evaluated directly with respect to the intended usage (Theis et al., 2015). In that spirit we evaluated how often our models produced affordances that were correct in the sense that they exactly matched test data with unseen objects. For a model $P_k(A|O)$ we define an accuracy measure as follows:

Algorithm 1 Accuracy computation of a model $P_k(A|O)$

procedure ACCURACY($P_k(A|O), l, m$,test_set)
  $s \leftarrow$ size(test_set)
  $x \leftarrow 0$
  for $(o_i, a_i) \in$ test_set **do**
    $A_o \leftarrow P_k(A|o_i)$ ▷ Output of the k-th model, sampled m times, with $m >> 1$
    SORT(A_o) ▷ The list of actions is sorted in descending order
    $sel_{il} \leftarrow$ FIRST(A_o, l) ▷ The most frequent actions up to l are kept
    if $a_i \in sel_{il}$ **then**
      $x \leftarrow x + 1$ ▷ x is increased when a_i is contained in sel_{il}
    end if
  end for
  $accuracy_k \leftarrow \frac{x}{s}$
end procedure

This measure tests how good a model replicates test data, and is meant to be a quantitative evaluation. Two different CVAEs are evaluated, the first with data encoded with GloVe-200 embeddings, the second with word2vec embeddings obtained over YAMC. We evaluated CVAE, K-NN and a baseline model by the described procedure. As baseline model we used a prior $P(A|O) = P(A)$, that is the probability distribution of actions over all objects. For every action a, $P(a) = N_a/N_{tot}$, where N_a is the number of times a appeared in the dataset. Accuracy computed on the test set for the different models are presented in Figure 2.

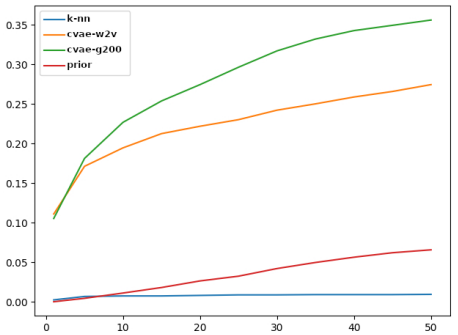

Figure 2: Computed accuracy for the different models. The X axis shows different percentages of retained output actions, starting from the most probable ones (parameter L). The Y axis shows the obtained accuracy.

The K-NN model fails to generalize the task: jumping to the closest object and outputting the empirical probability for it yield performances just above zero, also lower to the baseline.

We explain the K-NN performance as being this low due to the fact that similarity of objects (using cosine distance) does not encode similarity of associated actions. Supporting this hypothesis there is also the necessity of having nonlinear layers in the autoencoder in order to achieve high accuracy values. From this consideration we conclude that in word embedding space the mapping object-action is non-linear using the off-the-shelf embedding features.

The two CVAEs performance is higher, reaching a score of 0.35 with the off-the-shelf wordvectors. Additionally, we observed that training word2vec embeddings over the corpus lead to overfitting: performance computed over the test set comprising unseen objects is lower than the performance obtained with general purpose wordvectors.

6 Conclusions

With the goal of mining knowledge about affordance from corpora, we presented an unsupervised method that extracts object-action pairs from text using Semantic Role Labeling. The extracted pairs were used to train different models predicting $P(A|O)$: two Conditional Variational Autoencoders and one K-NN model. The presented results show that, on unseen objects, a CVAE trained on off-the-shelf wordvectors performs significantly better than the other tested models. Furthermore, we show how the K-NN model fails to generalize on our specific benchmark task, having performance even lower than the baseline model.

Knowledge about affordance, even in simple forms such as a object-action mapping, is relevant for applications such as inference of intent or robot planning. In robotics, planning requires a high amount of specifications inserted in the domain description, usually resulting in most of the decision rules being hand-crafted. With this paper, we present an algorithm allowing the leverage of knowledge about affordance present in corpora, thus allowing for a method of generating of at least a part the domain automatically.

Future work related to this research will be about improving the method by which the object-action pairs are mined, followed by reasearch on how this knowledge can be transformed to be used for robotic planning and intent recognition problems.

Acknowledgement

This work has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 721619 for the SOCRATES project.

References

Laura Antanas, Ozan Arkan Can, Jesse Davis, Luc De Raedt, Amy Loutfi, A. Persson, Alessandro Saffiotti, Emre Ünal, Deniz Yuret, and Pedro Zuidberg dos Martires. 2017. Relational symbol grounding through affordance learning : An overview of the reground project. In *International Workshop on Grounding Language Understanding (GLU). Stockholm, Sweden: Satellite of INTERSPEECH.*

Suna Bensch, Alexander Jevtić, and Thomas Hellström. 2017. On interaction quality in human-robot interaction. In *International Conference on Agents and Artificial Intelligence (ICAART)*, pages 182–189.

Elisheva Bonchek-Dokow and Gal A Kaminka. 2014. Towards computational models of intention detection and intention prediction. *Cognitive Systems Research*, 28:44–79.

Anna M Borghi, Andrea Flumini, Nikhilesh Natraj, and Lewis A Wheaton. 2012. One hand, two objects: Emergence of affordance in contexts. *Brain and cognition*, 80(1):64–73.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 10–21.

Xavier Carreras and Lluìs Márquez. 2004. Introduction to the conll-2004 shared task: Semantic role labeling.

Yu-Wei Chao, Zhan Wang, Rada Mihalcea, and Jia Deng. 2015. Mining semantic affordances of visual object categories. In *CVPR*, pages 4259–4267. IEEE Computer Society.

Haonan Chen, Hao Tan, Alan Kuntz, Mohit Bansal, and Ron Alterovitz. 2019. Enabling robots to understand incomplete natural language instructions using commonsense reasoning. *CoRR*.

François Chollet et al. 2015. https://keras.io Keras.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Nilgun Dag, Ilkay Atil, Sinan Kalkan, and Erol Sahin. 2010. Learning affordances for categorizing objects and their properties. In *2010 20th International Conference on Pattern Recognition*, pages 3089–3092. IEEE.

Dipanjan Das, Nathan Schneider, Desai Chen, and Noah A. Smith. 2010. Probabilistic frame-semantic parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 948–956, Stroudsburg, PA, USA. Association for Computational Linguistics.

Carl Doersch. 2016. Tutorial on variational autoencoders.

Aleksandr Drozd, Anna Gladkova, and Satoshi Matsuoka. 2016. Word embeddings, analogies, and machine learning: Beyond king-man+ woman= queen. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3519–3530.

Thomas E. Horton, Arpan Chakraborty, and Robert St. Amant. 2012. Affordances for robots: A brief survey. In *Avant. Pismo Awangardy Filozoficzno-Naukowej, 2*, volume 3, pages 70–84.

Gregory Finley, Stephanie Farmer, and Serguei Pakhomov. 2017. What analogies reveal about word vectors and their compositionality. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 1–11. Association for Computational Linguistics.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Comput. Linguist.*, 28(3):245–288.

James G Greeno. 1994. Gibson's affordances. *Psychological Review*, 101(2):336–342.

Thomas Hellström and Suna Bensch. 2018. Understandable robots - what, why, and how. *Paladyn, Journal of Behavioral Robotics*, 9(1).

Dan Hendrycks and Steven Basart. 2017. A quantitative measure of generative adversarial network distributions.

Peter Kaiser, Mike Lewis, Ronald PA Petrick, Tamim Asfour, and Mark Steedman. 2014. Extracting common sense knowledge from text for robot planning. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3749–3756. IEEE.

Ashutosh Kumar, Arijit Biswas, and Subhajit Sanyal. 2018. ecommercegan : A generative adversarial network for e-commerce. *arXiv preprint arXiv:1801.03244*.

Maya Çakmak Mehmet R. Doğar, Emre Uur, and Erol Şahin. 2007. Affordances as a framework for robot control. In *Proceedings of the 7th international conference on epigenetic robotics, epirob07*.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and James A. Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *In EMNLP*.

Michele Persiani, Alessio Mauro Franchi, and Giuseppina Gini. 2018. A working memory model improves cognitive control in agents and robots. *Cognitive Systems Research*, 51:1–13.

Ronald PA Petrick and Fahiem Bacchus. 2002. A knowledge-based approach to planning with incomplete information and sensing. In *AIPS*, volume 2, pages 212–222.

Alice Ruggeri and Luigi Di Caro. 2013. How affordances can rule the (computational) world. In *AIC@AI*IA*.

C. Schneider and J. Valacich. 2011. *Enhancing the Motivational Affordance of Human–Computer Interfaces in a Cross-Cultural Setting*, pages 271–278. Physica-Verlag HD, Heidelberg.

Alexander Sutherland, Suna Bensch, and Thomas Hellström. 2015. Inferring robot actions from verbal commands using shallow semantic parsing. In *Proceedings of the 17th International Conference on Artificial Intelligence ICAI'15*, pages 28–34.

Lucas Theis, Aäron van den Oord, and Matthias Bethge. 2015. A note on the evaluation of generative models. *CoRR*.

Natsuki Yamanobe, Weiwei Wan, Ixchel G Ramirez-Alpizar, Damien Petit, Tokuo Tsuji, Shuichi Akizuki, Manabu Hashimoto, Kazuyuki Nagata, and Kensuke Harada. 2017. A brief review of affordance in robotic manipulation research. *Advanced Robotics*, 31(19-20):1086–1101.

Philipp Zech, Simon Haller, Safoura Rezapour Lakani, Barry Ridge, Emre Ugur, and Justus Piater. 2017. Computational models of affordance in robotics: a taxonomy and systematic classification. *Adaptive Behavior*, 25(5):235–271.

Paper III

Traveling Drinksman—A Mobile Service Robot for People in Care-Homes

Michele Persiani, Çağatay Odabaşı, Florenz Graf, Mohit Kalra, Thomas Hellström, Birgit Graf

Accepted at the 52nd International Symposium of Robotics (ISR), 2020.

Traveling Drinksman — A Mobile Service Robot for People in Care-Homes

Michele Persiani[a], Çağatay Odabaşı[b], Florenz Graf[b], Mohit Kalra[b], Thomas Hellström[a], and Birgit Graf[b]
[a]Department of Computer Science, Umeå University, Umeå, Sweden
[b]Fraunhofer IPA, Stuttgart, Germany

Abstract

This paper describes ongoing work on the development of a service robot for serving drinks to people sitting at tables, for example in the recreation room of a care-house. The robot, denoted the *Traveling Drinksman*, should be able to detect the occupied tables, navigate safely according to defined policies, and interact with the humans sitting to serve them a drink. We present initial results addressing all of these problems with different sub-modules, including numerical results for the human detection module.

1 Introduction

With the globally growing elderly population foreseen for the coming decades, the need for infrastructure dedicated to elderly care has also increased [18]. As intelligent service robots become increasingly available, there will be many possibilities to delegate low-level, repetitive tasks to the robots, thereby giving human workers more time for interpersonal care.

Preliminaries user studies showed that robots can have a positive impact on elder population in care-house environments. In [19], the authors show how the introduction of a social robot strengthened the relationships between the guests and overall increased well-being. While also negative responses to socially assistive robots have been measured by e.g. the study in [12] (such as psychological factors emerging from the need of using assistive machines), the same study shows that, in general, there is a positive correlation between age and robot acceptance by the elders.

From a work perspective, it is still unclear the impact that service robotics will have on human labor [3], and most likely a balance between full automation and human-robot collaboration contains the sweet spot for service robotics. When seeing services as robot manufactured products, there is a fairly low threshold above which it becomes impossible for untrained human workers to resolve problems that might occur with the robot or with the provided services. Hence, fully automated service robots should keep a low level of complexity to avoid the constant need of expert personnel for the maintenance of the robots and their procedures.

Along these lines, this paper describes ongoing work on the development of a service robot for serving drinks, to be deployed for example in the recreation room of a care-house. However, the proposed solution can also in the future be extended towards more social aspects, such as entertainment or chit-chatting. The proposed robot - the *Traveling Drinksman* - is expected to take care of the drinking needs of several people in a room. The robot's task is to continuously detect and serve people at all occupied tables. We extend previously developed work [4] by implementing two major components: a human tracker, that allows to track in real-time the person positions in the room, and a planning system, that is used to plan how to serve the detected persons most efficiently.

Figure 1 The robot is approaching a human to ask if he wants to drink something.

2 Related Work

There is a fairly large body of research dedicated to service robotics, with one of the factors driving the field being the need for enhancing the infrastructure for the elderly care [18, 8]. In this regard, robotics offers a wide range of possibilities to support the working staff in e.g. care houses. Few examples of developed robots are nursing robots [7], companion robots [19], or robots for assistance in handicapped mobility scenarios [5, 17].

Due to their inherent complexity, robot architectures for service robotics should be able to be extended and to host newly added components systematically and comprehensively such that the provided services become easy to man-

age. The authors in [8] propose that languages such as the Unified Modeling Language are suitable to support such a requirement in architecture modularity, also for its suitability for system engineering methods.

Since our developed architecture provides a single service, we don't provide such a high-level description of the system and focus more on the implementation details. We use the Robot Operating System (ROS) [13], which is modular by construction and allows a seamless integration of components through their defined ROS API. ROS is a well-known robot development framework and is utilized by many projects. Other robot architecture frameworks are utilizable, such as ones developed in industry e.g. [7].

Two common problems in robotics that we also faced in this work are detecting persons and planning sequences of actions. Detecting humans can be necessary when the robot is working in shared spaces, and several solutions have been explored in past years using different combinations of sensors. For example using video [20], lidars [16], or sonars. Classification methods received an overall massive improvement in recent years due to the dawn of deep learning methods. Given a properly supervised dataset, these methods can provide reliable detections, also in real-time due to the parallelizable computations of neural networks. With this respect, we utilize a YoloV3 [15] classifier that provides bounding boxes for the trained image regions.

Planning is the task of finding the optimal sequence of actions that from initial conditions to achieve a desired goal condition. Many robotic applications using planners have been developed in past decades such as based on STRIPS [11], Hierarchical Task Networks, etc. For this work, we selected the Planning Domain Description Language (PDDL) [10], which is a standard language to specify planning domains for what is usually referred to as classical planning.

3 Method

We utilize an existing service robot prepared for the task of serving drinks [1]. The drinks are stored in plastic glasses in the robot's shelf-type hardware. The robot has an omnidirectional mobile base and uses a map to autonomously navigate collision-free through the environment. For making the robot approach and serve a given target position, we applied a previously developed approach module [4]. When triggered, this module makes the robot to approach a predefined area close to the person. By using an integrated tablet computer, users can request any of the available drinks stored inside the robot. In the proposed scenarios this procedure repeats until all present persons have been served.

We extend the functionalities offered by the approach module by new methods to increase autonomy, robustness and time efficiency. A robust human detection module (Section 3.1) and a global path planning optimization (Section 3.2) have been emerged as useful during previous tests in elderly care homes.

To realize the described robot behavior, we use the open-source software framework Robot Operating System (ROS) for development and testing. Figure 2 presents the three main parts of existing and new ROS modules, as well as their communication:

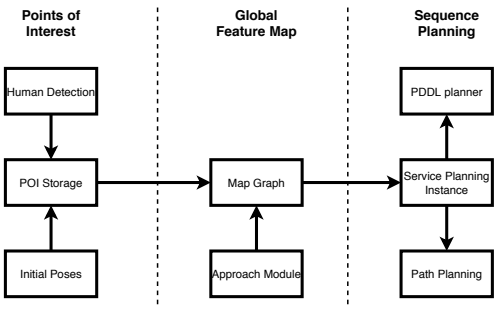

Figure 2 ROS software architecture of the high-level planning components of the system. Each block corresponds to a developed ROS node. Edges indicate how messages flow in between the nodes.

The first part provides the points of interest (POIs). In our case, these correspond to the person positions, the robot position, the glass refilling position, and the robot home position. Initially, these poses are set manually. The human detection simultaneously updates the person positions for approach and the next serving.

Human Detection: This ROS-node subscribes to the color and depth image of the robot for head detection. In the case of detection, the node provides the position of the head from the perspective of the camera. Using the transformation of the robot and from the localization, the node transforms the position into the global map frame.

Initial Positions: It provides the available sitting positions, the refilling, and home pose.

POI Storage: It publishes all the gathered detected positions. These include the person's global positions extracted from their head positions, the robot position, the sitting positions, the robot's home position, and the refilling position.

The second part of the architecture combines the global grid map of the robot with the POIs and the robot poses to approach these. Simultaneous Localization and Mapping provides the global grid map. Since the robot has to serve the drink close to the person, the approach node uses the costmap to provide the best not-occupied robot pose around the person pose. During the robot approaches a person, the approach module updates the best robot pose based on the current sensor information and human pose. This is required because of occluded and dynamic obstacles.

Approach Module: This module takes the pose of the person and returns the best robot pose for Robot Human Interaction as a ROS-Service.

Map Graph: Combines the robot map data with the approaching positions, the refilling and the home position.

The third part calculates the optimal overall sequence for serving the present people. First, the **Map Graph** node interconnects all POIs and store them as edges on a graph. Then, an external path planning service executes an A* search algorithm on the costmap to provide an estimated travel length. Afterward, we associate the travel length as cost and assign them to the edges. Finally, the PDDL-planner performs the overall planning, constraint by starting at the current robot position and ending at the refilling position. The PDDL-planner returns the optimal sequence of poses, that executes the state machine of the robot to serve all present person by a drink.

Path Planning: This node provides a path planning service using the global costmap map, which is used to compute the path length between two given sets of coordinates. The service request includes the positions and the service response includes the travel length.

Service Planning Instance: It is the node computing the service planning instance, that is updated in real-time using the global map. It allows us to plan for optimal paths over the feature map. For every plan request, a PDDL instance is generated using the feature map. The obtained plan is then transformed into sequences of approaches, refill, or go idle for the approach module. An edge associates every pair of nodes with associated weight obtained by calling **Path planning**.

PDDL Planner: This node provides a planner for PDDL problems. It is connected to the Map Graph to receive the graph and the travel lengths in between. After calculation, it returns the optimal sequence to the state machine.

More details on human detection and planning are given in the following sections.

3.1 Human Detection

In addition to traditional robot vision challenges such as obstacle detection, in our scenario, people are visible from all orientations. They may also be partly occluded, for example by a chair or other furniture. This causes problems for state-of-the-art approaches aiming at detecting the whole body. The other constraint is the need for 3D information. The robot needs to detect the precise position of the person in 3D space to approach him optimally. Therefore we decided to detect the head of people with an active stereo RGB-D camera that provides color and depth images.

Creating and hence finding a realistic RGB-D head dataset in uncontrolled environments is quite hard. Since the robot operates in uncontrolled environments like nursing homes, a deep network trained with a controlled dataset would not generalize to our problem well. That is why we start with the RGB-D human dataset [15] by Spinello et al., which contains thousands of full-body annotated RGB-D images from the people passing through a university hall. In the dataset, the people are mostly walking, standing, and their heads are visible from different angles to the camera. We extend it by annotating the head bounding boxes on both RGB and Depth domains.

We train two separate head detectors for RGB and Depth domains since both RGB and Depth domains have their advantages and disadvantages. The head detectors are object detection networks called YoloV3 [14]. RGB head detector is robust against sunlight but fails in the weak light. The depth detector works well in low light but is sensitive to sunlight due to the nature of the infrared-based depth camera.

The resulting RGB head detector is robust against sunlight but fails in the weak light. The depth detector works well in low light but is sensitive to sunlight due to the nature of the infrared-based depth camera. For these reasons, we fused the outputs of the two detectors to improve robustness. The fusion is done at the bounding box level, where the bounding boxes generated by the two modalities are fed into the SORT tracker [2], which then provides the fused detections.

Lastly, we transformed the head positions from the camera frame into the map frame to obtain all head detections in one frame for path planning. The head detection provides bounding boxes around the head block. To capture the distance of it, we calculate the median value inside the bounding box, since the assumption is that the head must be the most prominent and dominant object inside of the bounding box. Median filtering eliminates the background and provides us a simple, fast, and reasonably accurate estimation.

3.2 Planning

The robot has a map of the environment as well as the fixed table positions. The robot continuously updates the map with obstacles and persons by observing the environment with its available sensors [1]. As several people may be detected at the same time, a global planner determines the order in which the sitting positions should be served. Only sitting positions with at least a detected human are due for serving.

The defined problem corresponds to the Traveling Salesman Problem (TSP), which is the problem of visiting a set of points of interest while minimizing a cost metric (e.g. traveled path length), returning then to the starting point. In our case, there are additional constraints that the planner should consider, and for this reason, we used a general-purpose PDDL based planner, rather than a specific TSP algorithm. For example, only a limited number of glasses are available inside the robot, and plans should include also actions to refill the robot.

We selected Metric-FF [6] as a planner, which complies with PDDL 2.1. As the search strategy, we selected Dijkstra's algorithm. In our experiments the inclusion of a heuristics (i.e. turning the algorithm into A*) decreases the planning time but our tests showed that the planner also often returned sub-optimal plans. Our untested hypothesis is that the planner's available heuristics-based of deleting

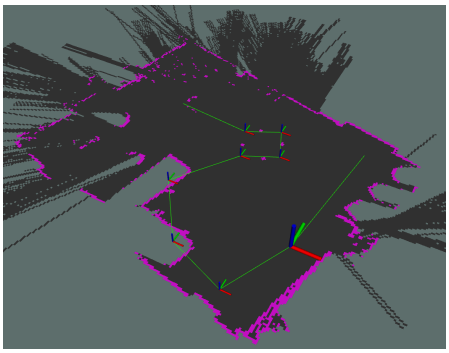

Figure 3 Obtained service plan for a scenario with 7 persons. From its initial position, the robot visits the refilling position, marked by the bigger axes, to then pass by every sitting position. Every plan ends with the robot reaching a predefined idle position. Service plans are then transformed into sequences of approaches for the approach module.

negative effects from the PDDL operators are sub-optimal for TSP planning instances (see Section 4).

A PDDL instance based on the map of the environment is continuously updated, and the planner is utilized to determine the path to follow in order to visit all currently detected occupied sitting positions, i.e. positions with at least a head detected in close proximity. While the goal of the generated plans is always the one of visiting all occupied sitting positions, the planning instance can be configured to minimize a chosen metric such as required time, traveled distance, a "First In First Served" policy, or a priority queue policy. The generated plans are dependent on the selected policy and correspond to an ordered list of sitting positions that is forwarded to the approaching module described in previous work [4]. The approaching module makes the robot approach the target position and start the serving procedure. After a position is approached, getting a drink or dismissing the robot through the tablet makes the robot continue to the next position in the plan.

4 Evaluation

We implemented planning between different tables in a real care house environment. Planning in a real scenario confirmed that our setting allows for the service robot to reach all of the tables in the most efficient way (by minimizing e.g. traveled length) from any position while avoiding obstacles. Obstacle avoidance is provided by the local planner. The robot plans also successfully consider refilling necessities by having the robot returning to its refilling station when the drink storage is empty.

A video of the robot executing a plan to visit the tables of a re-creation room at a care-house facility is at *https://youtu.be/hofICXrIvhE*. In the shown experiment, head tracking is disabled and the robot always visits all of the pre-programmed sitting positions.

We further compare the A* search strategy with Dijkstra

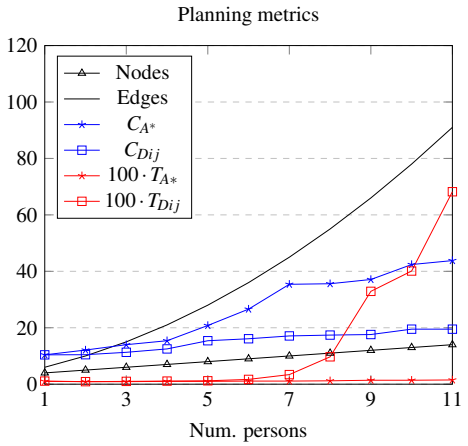

Figure 4 Metrics measurements for planning using the A* search strategy and Dijkstra's algorithm. C_{-} is the cost in meters of the obtained plans, T_{-} the plan time in seconds. Nodes and edges indicate the size of the planning instance.

Table 1 Average Precision (AP) for the RGB and Depth-based head detectors, computed on the test set for different IoU thresholds.

IoU	AP for RGB	AP for DEPTH
50%	0.90	0.89
60%	0.81	0.78
70%	0.68	0.58
80%	0.31	0.36

in scenarios of different sizes, up to 11 persons. If Figure 4 we show how A* provides plans that are in the most number of cases sub-optimal. On the other hand, the plans obtained using Dijkstra are always optimal but, as the graph shows, Dijkstra doesn't allow to scale to scenarios with many persons.

In order to test the human detector, we trained the head detector based on YoloV3 to detect people sitting at tables. The dataset utilized to train the classifier contains both RGB and Depth images. To compare and find the optimal modality, we trained different detectors for each modality. For training, we used 3043 RGB and Depth images, and for testing 351 images. Average Precision (AP) for the test set is presented in Table 1. Each row in the table shows the average precision (AP) for different values of *Intersection Over Union* (IoU). IoU quantifies how well the detector's predicted bounding box overlaps with the ground truth bounding box. The results show that both modalities work successfully.

The service robots always come to the proximity of the people, since they need to interact with them. The distance between humans and the robot would decrease less than 1m, and as the range reduces, the shadows in the depth images increases, which makes it harder to detect heads. Also, the dataset may not cover 360 degrees view of the head. Therefore, we create the following test bench. A

Table 2 Close up Head detection results on the robot under real conditions.

	# of Images	Correct Detection	Rate
Trial 1	133	39	0.29
Trial 2	89	28	0.31
Trial 3	80	42	0.525

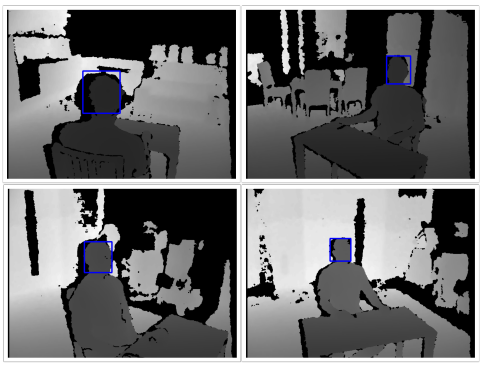

Figure 5 Success cases of the close-up head detection on the robot.

person is sitting on a chair in front of a table, as in Figure 5 and Figure 6. Then, the robot starts moving around the person so that it sees the head from every angle. The distance between the robot and the person is around 1m to 2m throughout the experiment. In Figure 5, there are blue bounding boxes around the head. These are the success cases as opposed to Figure 6.

There are three trials with two participants. In total, the robot collects 302 images. The results are presented in Table 2. As seen, the results are dramatically deviating from the dataset results. The foremost reason is that the head is too close to the camera limits, but at the same time, these are the situations that our service robots face during the care-house tests. That is why we need to test it against these situations as well. The main takeaway would be not to rely on single-frame detection but to rely on the fusion of detection over time.

The navigation stack on the robot calculates the accurate position of the robot by using three lidar sensors. We use this information to evaluate the depth estimation method. That is why we calculate the mean and standard deviation of the head position estimations through the time. The standard deviation on X-axis is 0.49 m and 0.34 m on Y-axis. Although these numbers seem huge, we need to consider that the detection experiments are pushing the limits of the camera, and there is an error caused by navigation. Furthermore, the depth estimation algorithm is a simple, fast, median value-based method. Also, this error can be compensated by an adaptive approaching algorithm presented in [4].

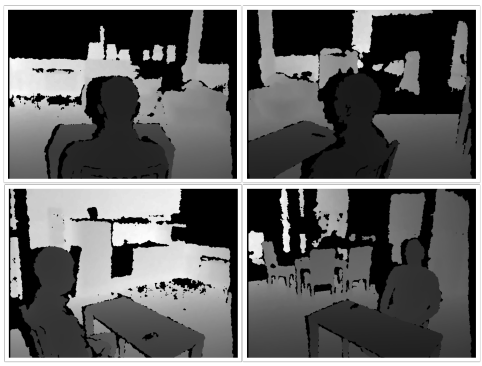

Figure 6 Failures of the close-up head detection on the robot.

5 Conclusions and Future Work

This paper describes ongoing work to design a service robot that efficiently serves drinks to the people in a room. We realized a human detection system based on depth and video cameras, and a planning systems suitable for the envisioned application. While their integration is still missing, we tested their performances separately, as shown in Section 4. The planning subsystem can be optimized in two ways. The first is by implementing a heuristic that is suitable for TSP problems. This would allow to use A* instead of Dijkstra, thus improving how the plan computation times scale in the number of graph nodes. The second way is to prune the search graph as right now the graph has several edges that scales quadratically in the number of nodes. Pruning the graph would allow to remove suboptimal edges, such as connecting nodes on the other side of the room with each other.

We tested the planning system in a real environment and verified its functioning. Preliminary results in real scenarios proved to be promising. However, at the moment the proposed system makes use of some simplifying assumptions. For instance, we explicitly assume that possible sitting locations are known and fixed. This is believed to be a realistic assumption since it is the case for most cafes, restaurants, and canteens. Future versions of the proposed system can remove this assumption to be able to serve also people standing at any position in the room. Furthermore, we identified cases where clustering persons and serving the clusters altogether can be a better solution rather than always serving individual persons. This can be the case when several persons are seated very closely. We also assume that no persons are walking during serving, and all persons in the room are sitting down by a table. Additional safety systems can be added to allow co-occupation of the space in the room between robot and persons, for example, [9] introduces six safety-related rules that the robot should follow when navigating spaces that can be occupied.

6 Literature

[1] Simon Baumgarten, Theo Jacobs, and Birgit Graf. 2018. The Robotic Service Assistant – Relieving the Nursing Staff of Workload. *ISR 2018 - 50th International Symposium on Robotics*.

[2] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. 2016. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*. 3464–3468.

[3] Michael Decker, Martin Fischer, and Ingrid Ott. 2017. Service Robotics and Human Labor: A first technology assessment of substitution and cooperation. *Robotics and Autonomous Systems* 87 (2017), 348–354.

[4] Florenz Graf, Çağatay Odabaşı, Theo Jacobs, Birgit Graf, and Thomas Födisch. 2019. MobiKa-Low-Cost Mobile Robot for Human-Robot Interaction. *International Symposium on Robot and Human Interactive Communication (RO-MAN), New Delhi* (2019).

[5] M Hans, B Graf, and RD Schraft. 2002. Robotic home assistant care-o-bot: Past-present-future. In *Proceedings. 11th IEEE International Workshop on Robot and Human Interactive Communication*. IEEE, 380–385.

[6] Jörg Hoffmann and Bernhard Nebel. 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research* 14 (2001), 253–302.

[7] Chandimal Jayawardena, I Han Kuo, Ulrike Unger, Aleksandar Igic, Richie Wong, Catherine I Watson, RQ Stafford, Elizabeth Broadbent, Priyesh Tiwari, Jim Warren, et al. 2010. Deployment of a service robot to help older people. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 5990–5995.

[8] Minseong Kim, Suntae Kim, Sooyong Park, Mun-Taek Choi, Munsang Kim, and Hassan Gomaa. 2009. Service robot for the elderly. *IEEE robotics & automation magazine* 16, 1 (2009), 34–45.

[9] Chi-Pang Lam, Chen-Tun Chou, Kuo-Hung Chiang, and Li-Chen Fu. 2010. Human-centered robot navigation—towards a harmoniously human–robot coexisting environment. *IEEE Transactions on Robotics* 27, 1 (2010), 99–112.

[10] Drew McDermott. 1998. PDDL-the planning domain definition language.

[11] Nils J Nilsson. 1984. *Shakey the robot*. Technical Report. SRI INTERNATIONAL MENLO PARK CA.

[12] Maribel Pino, Mélodie Boulay, François Jouen, and Anne Sophie Rigaud. 2015. "Are we ready for robots that care for us?" Attitudes and opinions of older adults toward socially assistive robots. *Frontiers in aging neuroscience* 7 (2015), 141.

[13] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. 2009. ROS: an open-source Robot Operating System. In *ICRA workshop on open source software*, Vol. 3. Kobe, Japan, 5.

[14] Joseph Redmon and Ali Farhadi. 2018. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* (2018).

[15] Luciano Spinello and Kai O Arras. 2011. People detection in RGB-D data. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 3838–3843.

[16] Tapio Taipalus and Juhana Ahtiainen. 2011. Human detection and tracking with knee-high mobile 2D LIDAR. In *2011 IEEE International Conference on Robotics and Biomimetics*. IEEE, 1672–1677.

[17] Tapio Taipalus and Kazuhiro Kosuge. 2005. Development of service robot for fetching objects in home environment. In *2005 International Symposium on Computational Intelligence in Robotics and Automation*. IEEE, 451–456.

[18] Department of Economic United Nations and Population Division Social Affairs. 2019. World Population Ageing 2019: Highlights.

[19] Kazuyoshi Wada and Takanori Shibata. 2007. Living with seal robots—its sociopsychological and physiological influences on the elderly at a care house. *IEEE transactions on robotics* 23, 5 (2007), 972–980.

[20] Torsten Wilhelm, Hans-Joachim Böhme, and Horst-Michael Gross. 2002. Sensor fusion for vision and sonar based people tracking on a mobile service robot. In *Proceedings of the International Workshop on Dynamic Perception*. 315–320.

Paper IV

Mediating Joint Intentions with a Dialogue Management System

Michele Persiani, Maitreyee Tewari

Accepted at 1st New Foundations for Human-Centered AI Workshop (NeHuAI), 2020.

Mediating Joint Intentions with a Dialogue Management System

Michele Persiani
Umeå University
Umeå, Sweden
michelep@cs.umu.se

Maitreyee Tewari
Umeå University
Umeå, Sweden
maittewa@cs.umu.se

ABSTRACT

A necessary skill which enables machines to take part in decision making processes with their users is the ability to participate in the mediation of joint intentions. This paper presents a formalisation of an architecture to create and mediate joint intentions with an artificial agent. The proposed system is loosely based on the framework of *we-intentions* and embodied on a combination of Plan Recognition techniques to identify the user intention, and a Reinforcement Learning network which learns how to best interact with the inferred intention.

KEYWORDS

Joint Intentions, Robotics, Goal Recognition, Reinforcement Learning

1 INTRODUCTION

The socio-technological evolution of human society has motivated the integration of robots in social and personal spaces. Hence, it is becoming a pressuring requirement for social robotics to understand human intentions and adapt to social values and needs.

Among other reasons, humans interact to understand and mediate intentions with other human participants [16]. A successful mediation of intention enable participants to decide profitable collaboration, to manage expectations, or to decide whether to trust the other participant. Natural language dialogues are among the primitive modes [4] of human-human interaction, and are also consistently used to mediate intentions. Dialogue management strategies have exploited joint intention theory for building team dialogues [15]. However, this work views joint intentions with an accent on joint task planning [13] for a human and a robot participant, rather than on the communicative protocols being involved.

The objective of this work is to model mediation of intentions for Human-Robot Interaction (HRI) in a household scenario, and is loosely based on the framework of *we-intentions* [17]. Within the scenario, we explore the cases where a person could need assistance from a robot such as: in cooking, finding different objects in the house, preparing for a visit to the supermarket, doctor or a friend. For instance, the person might say "I want to prepare a salad" to a robot, possibly having an intention for the robot to help her in cooking the dinner.

Hence, we explore the following research question: **how to create joint intention with machines?** The motivation behind this research question belongs to desired specifications of AI systems, including the need of an integrated cognition and collaboration mechanism, and a natural interaction between human and AI systems. Ultimately, it is about investigating the boundaries between the Eco-system of AI with that of human-beings.

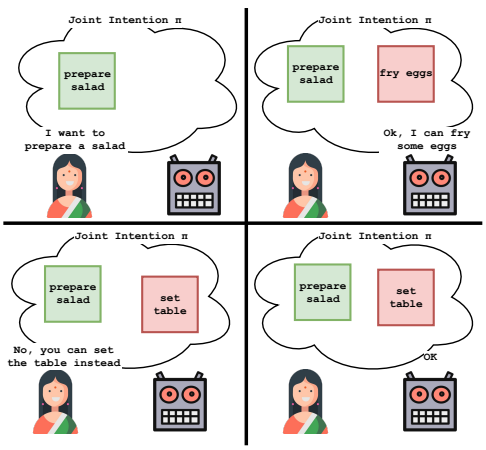

Figure 1: Creation of a joint intention with a robot. During its turn each participant adds and removes tasks (or primitive actions) from the shared intention π. Participants specify what they or the other will do until a common agreement is met.

To answer the research question, the formalisation of joint intentions in the context of shared task planning, and defining dialogue act functions [6] was done. This formalism offered a turn-based interaction scheme that allows two participant (human and a robot) to mediate an intention regarding a shared task.

2 METHODOLOGY

Some of the previous work [7, 11] proposed team rationality for building collaborative multi-agent systems, for example, in [11] the authors used Shared Plan [5] and *Propose Trees* to model collaboration as multi-agent planning problem, where a rational team will perform an action only if the benefits from performing an action is less than its cost. In [2] the authors formalised communication protocols using joint intention theory. The authors used joint persistent goals and persistent weak achievement goals to build joint intentions, and speech acts such as *request, offer, inform, confirm, refuse, acknowledge, and standing-offer* for their mediation.

As later described, we propose certain assumptions to lift some of the complexity that previous research utilizes in the—context of joint intention theory. We believe that such complexities, while theoretically sound, make implementations on real systems difficult and brittle; for this reason, we utilize a simplification of previous work's formalization for our needs. The rest of the section provides our simplified formalisation of mediating joint intention theory and attempts to briefly reason about the constraints posed.

Our proposed approach is based on predicate logic combined with planning, and is influenced by logic based semantics proposed in [1, 2, 14]. Agents are represented by $x, y, \dots x_1, x_2, \dots y_1, y_2, \dots$ and their actions by a_1, a_2, a_n. An intention of a single agent x is a plan $\pi = \{a_0, a_1, ..., a_n\}$ of actions together with a goal g the agent is committed to [16] and the intention is partially observed through $O \subseteq \pi$.

$Know(x, p) \equiv p \wedge Bel(x, p)$ represents the knowledge of agents and $MutBel(x, y, p)$ that x and y share a mutual belief about p—In our formulation an agent's intention is represented by the predicate $Intend(x, g, O)$ while a joint intention $JointIntend(x, y, g, O)$. An agent has an intention if following holds:

$$\begin{aligned} Intend(x, g, O) \equiv Know(x, \exists \pi\ O \subseteq \pi \wedge \\ Goal(\pi) = g \wedge \\ Commit(x, \pi)) \end{aligned} \tag{1}$$

i.e. not only is true that the agent has an intention and is committed to it, but the agent also has a belief about it. The set O is an explicit subset of π for which it is known that the agent already committed to it, and contains past observations or declarations about future commitments about π.

Eq. 2 expresses that to provide agents an intention doesn't require to explicit their full intention π, but only a part of it (see Figure 1), with the full intention being instead *inferred* by grounding the observed commitments in the task space.

A joint intention is an intention shared by the agents x and y with the same goal g. Therefore, a joint intention is a plan $\pi = \{a_{x0}, a_{y0}, a_{x1}, ..., a_{yn}\}$ together with a goal g where the actions in π can be allocated to either participants x or y. Furthermore, the involved agents have a mutual believe $MutBel$ about each others' commitments. Hence, two agents hold a joint intention if the following holds true:

$$\begin{aligned} JointIntend(x, y, g, O) \equiv Intend(x, g, O) \wedge Intend(y, g, O) \wedge \\ MutBel(x, y, JointIntend(x, y, g, O)) \end{aligned} \tag{2}$$

By this formulation x and y are allowed to have separate beliefs and inference mechanisms through which they find π; but are bound to have the same goal and observed commitments. Notice that this is a simplification of how joint intentions have been previously formalized in the literature, and to which we invite the reader. Nevertheless, this formalization is sufficient for our purpose of creating a dialogue manager that allows mediation of joint intentions. In this context of a dialogue between—two agents $\mathbf{x}$ and $\mathbf{y}$ we further make the following assumption:

$$\models \exists O \exists g\ JointIntend(\mathbf{x}, \mathbf{y}, g, O) \tag{3}$$

that translates as *there is always a joint intention between* $\boldsymbol{x}$ *and* $\boldsymbol{y}$. This assumption, while being quite strong, is quite reasonable for our context as the proposed DM is specifically tailored to mediate joint intentions. During every dialogue a joint intention is always obtained, and when the user leaves the conversation there is always an intention that was formed and is shared with the DM. Furthermore, it is always the case that the user utilises the DM to instantiate joint intentions. We do not take into consideration the cases in which the joint intention is bootstrapped or terminated as for example shown in [1].

Following the given definitions, we propose an interaction mechanism that allows two participants to collaboratively build O, by being able to add or remove actions from it. Currently, we have the following assumptions: 1) for every trial two participants are present, that is a human user and a Dialogue Manager (DM), that can be integrated for example in a house robot. 2) the DM is modelled to be user initiated, which always proposes the first action that will enter the set O.

Having an observed set O in a form of a partial plan, the DM can infer the most likely full intention π by utilizing plan recognition techniques as later described. This inference is based on the current state of the world that we assume to be available to the DM in the form of truth predicates. A possible architecture for maintaining an updated world description is not provided by this paper but can be for example implemented as in [3].

2.1 Goal Recognition and plan generation

At every turn of the dialogue the agent is required to infer the joint plan π to be able to participate in its mediation. For this purpose, we utilize plan recognition techniques based on the Planning Domain Definition Language (PDDL) [8]. PDDL belongs to the group of planning techniques known as classical planning, and allows to easily create non-hierarchical task domains.

For a given task domain we select the set of goals G as possible goals a user can pursue. Example of possible goals for Figure 1 can be to prepare dinner, lunch or breakfast. Plan recognition is achieved by a modified version of the method proposed in [10] with the following differences: 1) we allow the PDDL planner to plan using partially instantiated actions[1], and 2) the observations O are treated as a set rather than a sequence. Given an eventually empty set of observations O, goal recognition is performed as:

$$\hat{g} = \underset{g \in G}{argmax}\ \frac{C(\emptyset, g)}{C(O, g)} \tag{4}$$

where $C(O, g)$ is the cost of a plan achieving g and constrained to contain O, $C(\emptyset, g)$ is the cost of an optimal plan achieving g without being constrained by O. Hence, $0 \leq \frac{C(\emptyset, g)}{C(O, g)} \leq 1$ gives indication on how costly it is to deviate from an optimal plan achieving g for compliance with O. Finally, for an inferred goal $\hat{g}$ we obtain π as the optimal plan achieving g while being constrained to contain the observations O.

[1] We define a PDDL action as partially instantiated if not all of its arguments are grounded in the task domain. An action is fully instantiated when all arguments are grounded.

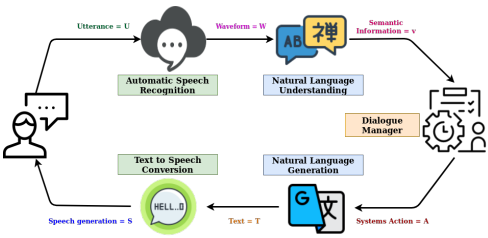

Figure 2: A traditional Spoken Dialogue Management System.

Dialogue Act	Precondition	Effect
$Offer_a, x, a$	$a \notin \theta \wedge a \notin R \wedge a \notin R$	$a \in \theta$
$Offer_g, x, \hat{g}$	$\emptyset$	$g = \hat{g}$
$Counter\text{-}offer, x, a_1, a_2$	$a_1 \in \theta \; a_2 \notin R \wedge a_2 \notin O$	$a_1 \notin \theta \wedge a_2 \in \theta$
$Accept, x, a$	$a \in \theta$	$a \notin \theta \wedge a \in O$
$Reject, x, a$	$a \in \theta$	$a \notin \theta \wedge a \in R$

Table 1: Speech acts for the SDS that allow to mediate actions with respect to the sets of offered, rejected and accepted commitments.

2.2 Mediation of Joint Intention

The agent and the user have to use a medium to communicate their joint intention, and to negotiate goals g and commitments O. In order to do that, we formalise a finite-state negotiation dialogue strategy with following dialogue acts: *(offer, counter-offer, accept, and reject)*. The dialogue strategies will be implemented with a spoken dialogue management system (SDS) 2.

Traditionally, an SDS consists of speech synthesis that recognises and generates speech, natural language understanding (NLU) transforms the human generated natural language to knowledge for the machine. A dialogue manager (DM) makes the decision based on the NLU and other components such as previous history, database etc, and natural language generation (NLG) receives the decision from the DM, transforms it to human understandable format and sends to speech synthesizer.

When the user generates its first utterance, it is transformed from speech to text and arrives at the NLU component. The NLU transforms the text to knowledge (semantic roles) and assigns a dialogue act *offer* [2]. An offer from the user instantiates the plan π by performing plan recognition, and creates a joint intention as described by *JointIntend*.

We define five dialogue acts $Offer_a$, $Offer_g$, *Counter-offer, Accept* and *Reject* with which both user and DM can mediate the intention's goal g and commitments O. Table 1 contains the effects of these dialogue acts with respect to three sets: θ is a set of offers, R and O are respectively the sets of rejected and accepted commitments. An $Offer_a$ represents offer about an action, $Offer_g$ indicates offer about a goal, *Counter-offer* is an action a_1 is not accepted and an alternative a_2 is instead proposed. An *Accept* and *Reject* can be used to accept or reject proposed commitments.

[2]At this stage we define a finite state SDS and only the user can initiate a dialogue only using *offer* together with a goal or an action.

Since a dialogue policy based on Finite-State-Machines is not realizable as it would need to consider all possible intentions, also based on the state of the task, we propose to learn the DM dialogue policy with Reinforcement Learning methods. This approach is not new in the context of dialogue management, and by this method the user is simulated by an Agenda [12].

2.3 Learning the agent strategy with Reinforcement Learning (RL)

At every turn, a Q-Network [18] evaluates the current inferred π together with the actions in the sets θ, R and O and the current PDDL state, selecting which dialogue act to perform by an ϵ-greedy policy computed on the dialogue acts expected return. In RL, agents learn which policy to adopt by maximising the reward they receive during each episode. The current version of the reward function is:

$$R = -\alpha T + \beta \frac{\overline{\pi} \cap \pi}{\overline{\pi} \cup \pi} + \gamma \frac{C(\overline{\pi}, \overline{g})}{C(\pi, \overline{g})} \tag{5}$$

where $\overline{\pi}$ and $\overline{g}$ form the user's original desired joint intention (held by the Agenda). The first terms penalises every turn that the interaction takes, hence making interactions as short as possible. The second term evaluates how the final resulting intention is similar to the one the user had as objective for the interaction. The third term evaluates instead the cost the resulting mediated intention has, compared to the user's original one. α, β and γ determine how the three components of the reward function are weighted. Notice that the system cannot access $\overline{\pi}$ and $\overline{g}$, that are instead only used at the end of every interaction for evaluation. Thus, the the DM learns to mediate and improve the unobservable user intention $\overline{\pi}, \overline{g}$.

3 FUTURE WORK

The research is still in its early stages and we are currently implementing the described system. We developed the goal recognition and the reinforcement learning components together with a simple user Agenda. The Agenda is based on PDDL and simulates how the user would modify the joint intention during its turn, while having as objective a randomly generated joint intention.

Initial experiments gave positive results, in the sense that the RL is able to learn the structure of the problem for simple scenarios, and successfully maximises the possible rewards. Several investigations are needed and are still open: what is the Q-Network learning? Does our current setting allows any generalisation? The current implementation requires hundreds of episodes to converge. Can

the process be made faster/simpler? How to facilitate the online adaptation over real users?

Encapsulation of the joint intention model with SDS is still to be implemented. For early prototypes of the system we plan to implement the dialogue manager as described in Section 2.2. Later versions could see the implementation of a more complete SDS through for example a POMDP model [9]. This could allow to have dialogues that are not strictly related to the mediation of the joint intention, but rather more flexible and intuitive for the user. Finally, investigation about the soundness of this approach in real scenarios for example in user studies is still to be performed.

ACKNOWLEDGMENTS

This work has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 721619 for the SOCRATES project.

REFERENCES

[1] Philip R Cohen. 2019. Foundations of Collaborative Task-Oriented Dialogue: What's in a Slot?. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*. 198–209.

[2] Philip R Cohen and Hector J Levesque. 1990. Intention is choice with commitment. *Artificial intelligence* 42, 2-3 (1990), 213–261.

[3] Sandra Devin and Rachid Alami. 2016. An implemented theory of mind to improve human-robot shared plans execution. In *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 319–326.

[4] New World Encyclopedia. 2017. Dialogue — New World Encyclopedia,. //www.newworldencyclopedia.org/p/index.php?title=Dialogue&oldid=1007366 [Online; accessed 24-January-2020].

[5] Barbara Grosz and Sarit Kraus. 1996. Collaborative plans for complex group action. *Artificial Intelligence* (1996).

[6] Bunt Harry, Petukhova Volha, Traum David, and Alexandersson Jan. 2017. *Dialogue Act Annotation with the ISO 24617-2 Standard*. Springer International Publishing, 109–135.

[7] Ece Kamar, Ya'akov Gal, and Barbara J Grosz. 2009. Incorporating helpful behavior into collaborative planning. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. Springer Verlag.

[8] Drew McDermott. 2003. The Formal Semantics of Processes in PDDL. In *Proc. ICAPS Workshop on PDDL*. ICAPS Workshop, Trento, Italy, 8.

[9] Michael Frederick McTear, Zoraida Callejas, and David Griol. 2016. *The conversational interface*. Vol. 6. Springer, Spain.

[10] Miguel Ramírez and Hector Geffner. 2010. Probabilistic plan recognition using off-the-shelf classical planners. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*. MIT Press, USA, 6.

[11] Timothy W. Rauenbusch and Barbara J. Grosz. 2003. A Decision Making Procedure for Collaborative Planning. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '03)*. Association for Computing Machinery, 1106–1107.

[12] Jost Schatzmann, Blaise Thomson, and Steve Young. 2007. Statistical user simulation with a hidden agenda. In *Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue*. 273–282.

[13] David P. Schweikard and Hans Bernhard Schmid. 2013. Collective Intentionality. In *The Stanford Encyclopedia of Philosophy* (summer 2013 ed.), Edward N. Zalta (Ed.). Metaphysics Research Lab, Stanford University, USA.

[14] Ira A Smith, Philip R Cohen, Jeffrey M Bradshaw, Mark Greaves, and Heather Holmback. 1998. Designing conversation policies using joint intention theory. In *Proceedings International Conference on Multi Agent Systems (Cat. No. 98EX160)*. IEEE, 269–276.

[15] Rajah Subramanian, Sanjeev Kumar, and Philip Cohen. 2006. Integrating Joint Intention Theory, Belief Reasoning, and Communicative Action for Generating Team-Oriented Dialogue.. In *Proceedings of the National Conference on Artificial Intelligence*, Vol. 21. MIT Press, Boston, USA, 6.

[16] Michael Tomasello, Malinda Carpenter, Josep Call, Tanya Behne, and Henrike Moll. 2005. Understanding and sharing intentions: The origins of cultural cognition. *Behavioral and Brain Sciences* 28, 5 (2005), 675–691.

[17] Raimo Tuomela. 2005. We-intentions revisited. *Philosophical Studies* 125, 3 (2005), 327–369.

[18] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double q-learning. In *Thirtieth AAAI conference on artificial intelligence*. MIT Press, Arizona, USA, 7.