MASTER'S THESIS | LUND UNIVERSITY 2015

Monitoring and Managing Interaction Patterns in Human-Robot Interaction

Felip Martí Carrillo

Department of Computer Science
Faculty of Engineering LTH

ISSN 1650-2884
LU-CS-EX 2015-37

Monitoring and Managing Interaction Patterns in Human-Robot Interaction

Felip Martí Carrillo
felip@openmailbox.org

4th September, 2015

Master's thesis work carried out at the Department of Computer Science, Lund University.

Supervisor: Elin Anna Topp, elin_anna.topp@cs.lth.se

Examiner: Jacek Malec, jacek.malec@cs.lth.se

Abstract

Nowadays, one of the most challenging problems in Human-Robot Interaction (HRI) is to make robots able to understand humans to successfully accomplish tasks in human environments. HRI has a very different role in all the robotics fields. While autonomous robots do not require a complex HRI system, it is of vital importance for service robots.

The goal of this thesis is to study if behavioural patterns that users unconsciously apply when interacting with a robot can be useful to recognise the users' intentions in a particular situation.

To carry out this study a prototype has been developed to test in an automatic and objective way, if those interaction patterns performed by several users in the area of service robots are useful to recognise their intentions and disambiguate unclear situations.

Keywords: Interaction Patterns, Human-Robot Interaction, Service robots, Robot learning, ROS, Bayesian Network

Acknowledgements

First of all, I would like to express my gratitude to my supervisor Elin Anna Topp for her support, help and guiding from the very beginning, when replying an e-mail to an unknown student from Barcelona who was asking for a Master Thesis Project, until the end of the project.

I also would like to thank all those who have been my family in Lund and have been part of my amazing experience abroad: my corridor mates (9B våning fyra), international friends, and my mentors (EC/DC).

And last but not least, I would like to thank my family and friends from Barcelona for their support and comprehension.

Contents

Chapter 1
Introduction

In this introductory chapter we present our motivations for doing this project. Besides, we define the problem that we want to solve and the contributions.

1.1 Motivation

The median age of the world's population is increasing, in some regions much more than others such as in Europe or Japan. This situation has led some governments to invest in service robotics research. More specifically, research focused on service robots in assisted living environments which can help an ageing population to remain active and independent for longer.

One of the most important aspects of the service robots is the Human-Robot Interaction. The robot has to be able to understand the human, and the other way around as well. Humans are unpredictable, there are a lot of ways to show an object, a region, a task, or a workspace. Therefore, robots have to be intelligent and try to understand humans taking into account all the inputs that they receive.

Human-Robot Interaction and Human activity recognition are not only specific for service robots. Actually, they are being used in all robotics fields, even in industrial robots with the idea of direct interaction and easier programming. If industrial robots were provided with a cognitive system, robots would be able to understand different situations and also will be able to learn new tasks from users. Consequently, industrial robots will be more flexible in use, and it will reduce time and cost for small and medium-sized enterprises (SMEs) with small lot-sizes.

The robot cognition and the interaction with robots and users are of great interest. Due to its applicability to all of the robotics areas, the research progress and results done in service robots could be transferred to another robotic division.

Lund University is part of the consortium SMErobotics (`www.smerobotics.org`) where the involved departments (Department of Automatic Control, Department of Computer Science and Department of Mathematics) provide their relevant expertise.

Therefore, it seems reasonable to investigate the possibilities of transferring advances from this project to the industrial setting investigated in the SMErobotics project, where an ongoing study aims to investigating recognisable actions.

1.2 Problem Definition

In this project we focus our investigation on trying to provide basic knowledge for a robotic system to make it able to understand the overall situation and possibly recognise the user's intentions in a particular situation. This particular situation is a "guided tour" performed by several users with a mobile robot in a particular part of the Computer Science department at Lund University.

Unclear situations can always appear in any human-robot communication context. Therefore, if a robot is provided with basic knowledge, it will be able to solve this kind of ambiguities or at least will be able to take the initiative asking the human for more information to clarify the context.

To do so, we are going to work with the idea of interaction patterns. Interaction patterns are behavioural patterns that the user unconsciously applies when interacting with the robot [31]. Those patterns comprise features like commands given to the robot, certain movements, activity chains preceding a certain utterance or activity, and can be applied to disambiguate explicitly (e.g., verbally) given instructions or information.

Our approach is to use interaction patterns occurring around every object, workspace and region presentation made by the user in a familiar environment [30]. With these interaction patterns we want to use a suitable machine learning algorithm to train, and later on perform inference with new data in order to recognise the users' intentions or unclear situations.

To carry out our study and check the proposed approach we are going to implement a software prototype. This prototype will consist of an interaction monitor module that has to consider several different sources of data to generate an understanding of the overall situation.

1.3 Contribution

In this project we contribute modestly in the Human-Robot Interaction field, trying to understand, and recognise unclear situations generated by the user in the interaction with a service robot.

More specifically, our contribution is the confirmation of the hypothesis presented about Spatial Concepts from User Actions [31] in an objective and automatic way. The hypothesis suggested that there are certain patterns observable, consisting of combinations of user movements and actions, that can be used for confirmation or detection of ambiguities.

We can use users' interaction patterns while interacting with a robot in order to recognise and understand their intentions.

1.4 Thesis Organization

This report is structured in 5 chapters and 1 appendix.

Chapter 2 - Background

Chapter 2 gives an overview of relevant work that can be related to the work presented in this thesis. Furthermore, this chapter presents a theoretical background that will be used for our approach.

Chapter 3 - Approach

The description of our solution and the structure of the software prototype implemented can be found in this chapter.

Chapter 4 - Evaluation

In chapter 4 is explained which kind of tests have been done and which are the results obtained in order to evaluate our approach.

Chapter 5 - Conclusions

Chapter 5 presents the conclusions that can be drawn from the results of our study. In addition, some future work and ideas are summarized.

Appendix A

Instructions on how to install and run the prototype implemented can be found in this appendix.

Chapter 2
Background

The state of the art is presented in this chapter considering different, but related robotics fields. Besides, we introduce a theoretical background of the techniques used for the prototype.

2.1 Related work

In the field of service robotics, Human-Robot Interaction has by now become quite an established area of research. There has been some research about how people modify their behaviour depending on their interaction partner's needs and understanding [6] [13] [4], in the case of robots depending on their feedback [33].

Working under the premise that a human-like body will provide an abundance of non-verbal information useful to smoothly communicate with the robot, Kanda et al. [14] developed an interactive humanoid robot to evaluate the body movement interaction between the humanoid robot and humans.

Spexard et al. [27] combined different interaction concepts and perception capabilities integrated on a humanoid robot to achieve comprehending human-oriented interaction. They bring together people tracking [17], face and voice detection [32]. Multimodal cues such as gazing direction, deictic gestures [10], and the mood of a person are also considered as an input for human-oriented interaction.

Hüttenrauch et al. performed a study with 22 subjects to investigate the spatial distances and orientations of users interacting with a robot [11].

Ross Mead and Maja J Matarić [19] also worked in the multimodal communication for social robots. They studied how to adapt the distance, orientation, speech and gesture between two social agents.

The CORAL research group also studied Human-Robot Interaction focusing on how people that are not supervising a robot can make more accurate inferences about the robot's state [24]. This work is based on the idea of robots asking for help when they detect uncertainty or unclear situations, instead of humans continuously supervising robots to detect this kind of situations [25].

Additionally, they have developed a receptionist robot that is able to recognise gestures using an RGB-D camera. This robot is able to interact successfully with people considering the user orientation and proximity respect to the robot, and the gesture performed by the user [15]. However, the robot uses a specific algorithm to recognise each gesture, and cannot disambiguate unclear user intentions.

To recognise or classify the users' intentions, we are going to use a machine learning algorithm. A Bayesian network is a machine learning algorithm that has been used to classify personality traits based on tactile interaction patterns with a robot [12]. In this paper they obtain successful results, classifying introverted and extroverted people using a General Bayesian Network (GBN) [18]. The tactile interaction patterns were coded into two items: touched location and type of touch.

Bayesian networks are also used by Glas et al. [9] to predict a discrete shopkeeper robot action based on the behaviour of a costumer.

For industrial human-robot interaction scenarios, a two-step approach for activity recognition based on skeleton features is presented in this paper [23]. They use an RGB-D camera to obtain raw skeleton data, and they also use Random Forests and Hidden Markov Models (HMM) to classify three groups of activities: Movement, Gestures, and Object handling.

Hidden Markov Models are also used by Christopher Lee and Yangsheng Xu [16] to interactively recognize sign language alphabet gestures. They perform online learning of new gestures after only one or two examples of each.

Finally, this master thesis is based on the previous work done by Elin Anna Topp in a user study regarding particular observable "interaction patterns" in the interaction during a "guided tour" [31]. Here, after studying different people guiding and presenting the office environment to a service robot, it was concluded that there are certain observable "patterns" that can provide extra information to clarify mismatches.

2.2 Theoretical background

To evaluate and test our approach we are going to implement a software prototype. This prototype will be explained in more detail in chapter 3. However, in this section we are going to introduce a basic theoretical background used in our approach.

We are going to work with the idea of **Interaction Patterns** performed by users presenting places and objects (**Spatial Concepts**) in a familiar environment in order to recognise with a **Machine Learning algorithm** the user's intentions or to find unclear situations.

As aforementioned, this study is based on a previous one [31] in which there are some text files with annotations that will be **parsed** in order to extract all the data. Furthermore, we are going to integrate those modules with **ROS**.

2.2.1 Spatial Concepts

A previous study about Human Augmented Mapping [30] used a hierarchical structure for the space representation, in which two main categories are used:

- **Location:** Specific positions/areas that can represent the position of large objects that are considered static. For example a coffee machine, a refrigerator or a printer.
- **Region:** Any portion of space that is large enough to allow for different locations in it, or at least large enough to navigate in it. Typically this would be rooms, corridors or parts of those.

In this project we refer to the location category as a **Workspace**. Those two categories are used to represent places on a map. However, there is another category: **Object**, not used for mapping, but very important as well.

2.2.2 Interaction Patterns

Interaction Patterns are a set of behavioural features that the user unconsciously applies when interacting with a robot. Some examples of behavioural features could be some gestures such as to point, touch, or grab an object; some commands given to the robot; or even some movements performed with respect to the robot.

With a set of behavioural features we have a behavioural pattern, or an interaction pattern. It was previously studied that those patterns are different depending on the thing that is being presented to the robot [31]. Furthermore, those patterns are quite common independently of the subject that is interacting with the robot.

2.2.3 Machine Learning Algorithms

We are going to use a machine learning algorithm in order to classify different categories. Therefore, in this section we present several algorithms that can be used as classifiers.

Artificial neural network

Artificial neural network (ANN) is a flexible mathematical structure inspired by the functionality of the human brain. It has been successfully applied in industrial applications such as statistical pattern recognition, classification, identification, control, modelling, etc. [20]

ANN consists of a set of nodes (artificial neurons) in which each node represents a mathematical function. Those nodes are interconnected by direct links in order to activate other nodes. Each link has a numeric weight, which determines the strength and sign of the connection [26].

The network is structured by an input layer of nodes, some hidden layers, and finally by the output layer (Example in figure 2.1). The number of input nodes is typically taken to be the same as the number of input variables. The number of output nodes is typically the number that identifies the general category of the state of the system. A trial-and-error approach is usually used to determine the number of hidden layers [20].

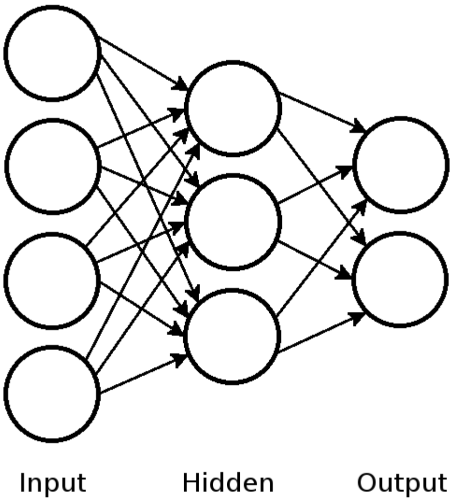

Figure 2.1: Artificial Neural Network with 4 input nodes, 1 hidden layer with 3 nodes, and 2 output nodes

Bayesian network

A Bayesian network, also known as belief network, probabilistic network or knowledge map is a kind of probabilistic graphical model for data structures that represent conditional dependencies among a set of random variables through a direct acyclic graph (DAG) [26].

A graph is represented by nodes (vertices), and links (arcs). In the case of Bayesian networks nodes represent random variables and are drawn as circles labeled with the variable name. The links, which have a particular directionality indicated by arrows, represent direct dependencies among these variables, and are drawn with arrows between nodes [3].

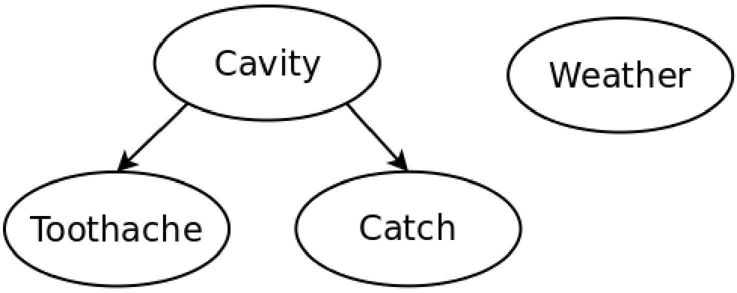

Figure 2.2: Bayesian network with 4 variables and their corresponding relationships

In the figure 2.2 a Bayesian network is represented with 4 different variables `Cavity, Weather, Toothache` and `Catch`. `Weather` is a variable independent of the others, so it has no links to other variables. Given a `Cavity`, `Toothache` and `Catch` are conditionally independent variables, therefore there are no links between `Toothache` and `Catch`. However, `Cavity` is a direct cause of `Toothache` and `Catch`, so an arrow is drawn for each direct dependency.

Decision Tree Learning

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences. It is a flowchart-like structure in which an internal node represents a test on an attribute value, each branch represents an outcome of the test, and each leaf node represents a class label, which represents the decision taken after following the outcome of each test to the leaf node. A path from the root node to a leaf node may represent a classification rule [21].

The decision tree learning algorithm adopts a greedy algorithm strategy. It always tests the most relevant, important, or the best attribute talking in terms of efficiency in finding the correct classification. This test divides the problem up into smaller subproblems that can then be solved recursively. That way, we hope to get to the correct classification with a small number of tests, meaning that all paths in the tree will be short and the tree as a whole will be shallow [26].

Gaussian Mixture Model

A mixture model is a probabilistic model for representing the presence of subpopulations within an overall population. The Gaussian Mixture Model (GMM) [34] is a finite mixture probability distribution model that is generated from a finite number of Gaussian distributions with unknown parameters. Those parameters are set during the learning process depending on the algorithm used.

Figure 2.3 shows the classification of different kind of transports using GMM, in which there are 4 mismatches. The classifier uses the Expectation Maximization algorithm for fitting mixture of Gaussian models.

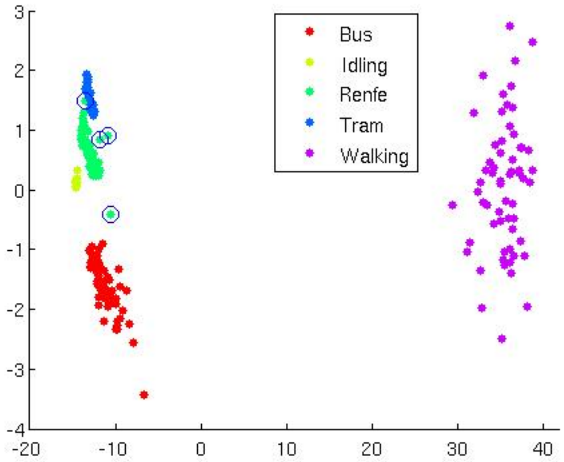

Figure 2.3: Classification using Gaussian Mixture Models

Random Forests

Bagging or bootstrap aggregation is a technique for reducing the variance of an estimated prediction function. Bagging seems to work especially well for high-variance, low-bias procedures, such as trees. The essential idea in bagging is to average many noisy but approximately unbiased models, and hence reduce the variance [8].

Random forest is based on the idea of bagging. It is a classifier consisting of a collection of tree-structured classifiers where there are independent identically distributed random vectors. Each tree casts a unit vote for the most popular class at one input [5].

2.2.4 Parsing

Parsing is the process of structuring an input sequence (for instance from a keyboard or a file) in accordance with a given grammar. A parser transforms input strings into a data structure, generally in a tree structure, that reflects the implicit hierarchical structure of the text and allows a posterior precessing.

In the case of markup languages such as XML or HTML a parser is used as the file reading facility of a program. The parser reads the XML or HTML tags in a file to obtain the data contained in those tags.

Parsing ELAN files

For this project, it will be necessary to parse some ELAN annotation files. ELAN [1] is a professional tool for the creation of complex annotations on video and audio resources. An ELAN annotation file is as an XML file with the following format:

```
<?xml version="1.0" encoding="UTF-8"?>
<ANNOTATION_DOCUMENT AUTHOR="" DATE="2015-03-05T15:45:49+01:00"
                     FORMAT="2.8" VERSION="2.8">
    <HEADER MEDIA_FILE="" TIME_UNITS="milliseconds">
    </HEADER>
    <TIME_ORDER>
        <TIME_SLOT TIME_SLOT_ID="ts1" TIME_VALUE="50"/>
        <TIME_SLOT TIME_SLOT_ID="ts2" TIME_VALUE="50"/>
        ...
        <TIME_SLOT TIME_SLOT_ID="ts774" TIME_VALUE="1398080"/>
    </TIME_ORDER>
    <TIER DEFAULT_LOCALE="en" LINGUISTIC_TYPE_REF="speech"
          TIER_ID="rob_sp">
        <ANNOTATION>
            <ALIGNABLE_ANNOTATION ANNOTATION_ID="a389"
                                  TIME_SLOT_REF1="ts1"
                                  TIME_SLOT_REF2="ts3">
                <ANNOTATION_VALUE>I am ready to go</ANNOTATION_VALUE>
            </ALIGNABLE_ANNOTATION>
        </ANNOTATION>
        ...
    </TIER>
    ...
</ANNOTATION_DOCUMENT>
```

An annotation document mainly has two important tags:

- **TIME_ORDER** is a list of time slots where the time is specified in milliseconds
- **TIER** is a list of different kind of annotations `usr_mov, rob_mov, rob_sp, usr_gesture, usr_present, etc.` In each annotation we can find the initial and final time reference, together with the annotation text. Annotations that are relevant to our case study will be explained further in section 3.1.2.

Figure 2.4 shows a graphical representation of a parsed ELAN file.

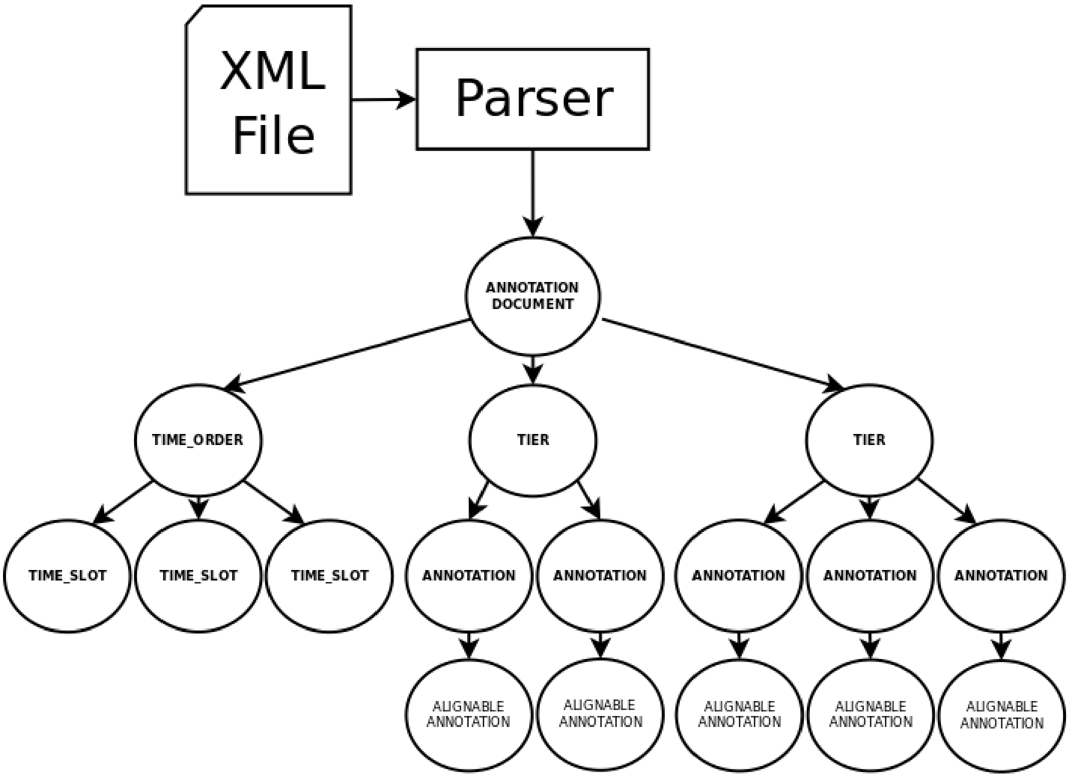

Figure 2.4: Hierarchical structure of an ELAN annotation file parsed

2.2.5 ROS

The Robot Operating System (ROS) is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms [7].

Indigo Igloo is the 8th official ROS release that came out in July 2014. It is supported on Ubuntu 13.10 (Saucy Salamander) and Ubuntu 14.04 LTS (Trusty Tahr). Furthermore, ROS Indigo is the version used for the implementation of this prototype, using catkin as a build system.

Further, in chapter 3 we will explain the implementation of the prototype in ROS. Therefore, we are going to introduce a general basic idea about the software structure and the kinds of communication in ROS.

ROS software structure

- **Packages:** Software in ROS is organized in packages. A package might contain ROS nodes, a ROS-independent library, a dataset, configuration files, a third-party piece of software, or anything else that logically constitutes a useful module.

- **Metapackages:** A metapackage simply references one or more related packages which are loosely grouped together. Metapackages are specialized Packages in ROS (and catkin). They do not install files (other than their package.xml manifest) and they do not contain any tests, code, files, or other items usually found in packages.

- **Node:** A node is a process that performs computation. Nodes are combined together into a graph and communicate with one another using streaming topics, Remote Procedure Call (RPC) services, and the Parameter Server. These nodes are meant to operate at a fine-grained scale; a robot control system will usually comprise many nodes. For example, one node controls a laser range-finder, one Node controls the robot's wheel motors, one node performs localization, one node performs path planning, one node provide a graphical view of the system, and so on.

ROS communication structure

- **Topics:** are named buses over which nodes exchange messages. Topics have anonymous publish/subscribe semantics, which decouples the production of information from its consumption. In general, nodes are not aware of who they are communicating with. Instead, nodes that are interested in data `subscribe` to the relevant topic; nodes that generate data `publish` to the relevant topic. There can be multiple publishers and subscribers to a topic.

- **Services:** Topics are intended for unidirectional streaming communication. Nodes that need to perform remote procedure calls, i.e. receive a response to a request, should use services instead.

Chapter 3
Approach

Our approach is to obtain users' interaction patterns when interacting with a service robot from different sources of data to generate an understanding of the overall situation. As aforementioned, this thesis is based on the study case "Understanding Spatial Concepts from User Actions" [31]. Therefore, all the data that we are going to use come from ELAN [1] annotation files and the recordings of a laser scan used in that study.

However, we have implemented our approach considering future sources of data. The main idea of the proposed design is to be able to use this prototype in live without using preprocessed annotation files with observations of the users' behavioural features.

The prototype is divided in two parts. The first one is the **Interaction Patterns Manager** which is the main core of our approach. This module is the one who stores from different sources all the user's behavioural features that occur around every object, workspace or region presentation, and it tries to recognise what the user is presenting or showing to the robot.

The second module is the **User movements from the Tracker** which will be the first step in obtaining automatically data from a source that is not an annotation file. This module extracts all the users' behavioural features from trajectories generated by a tracker [29]. Those trajectories are generated using data from the service robot laser scanner.

Even though the prototype is divided in two modules that are not fully connected, it has been implemented considering an easy way to connect all the parts. A future contribution will be a gesture recognition module using a kinect camera that can be added to the service robot.

Moreover, it is assumed that the results of this project can be transferred to the industrial setting investigated in the SMErobotics project. Robot Operative System (ROS) [7] is the robotic framework standard for the interaction studies within SMErobotics because of its flexibility and availability.

Therefore, the prototype has been implemented in ROS and in C++. Five different ROS packages in the first module and one more in the second module have been developed in order to obtain a generic and a modular design.

3.1 Interaction Patterns Manager

The implementation of this part of the prototype is based on annotation files generated with the respective tool ELAN [1], from videos recorded during the study case "Understanding Spatial Concepts from User Actions" [31]. Those annotation patterns were identified and confirmed in a manual analysis effort.

Figure 3.1 shows all nodes and topics of this prototype. It consists of a **parser** to obtain all the annotations from ELAN files; an **ELAN translation** module to translate Interaction Patterns from annotations (Strings) to variables adapted for the Machine Learning algorithm (Integers); an **interaction monitor** that store all the immediate around Interaction Patterns and send it to the **interaction learner** in the case that we want to train the algorithm; or to the **interaction recognition** in the case that we want to recognise the category of the item that the user is presenting.

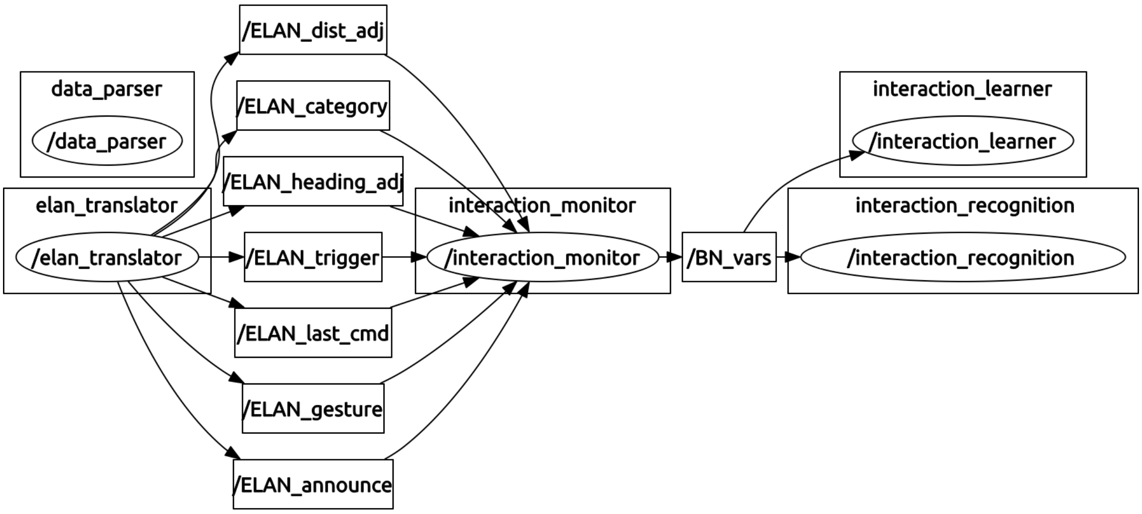

Figure 3.1: Interaction Patterns Manager prototype nodes and topics structure

3.1.1 Machine Learning Algorithm

In section 2.2.3 we have presented five different machine learning algorithms that could be used for our approach. However, we are going to use only one, and we have to choose the most suitable one.

Algorithms that work with a tree structure such as Decision Tree Learning and Random Forests have the inconvenience of the tree design, a tree structure has to be designed beforehand. We have a bunch of data that was previously studied, but not at the level of designing a decision tree. Furthermore, taking into account all the different variables, the design of the tree would require a big effort.

Gaussian Mixure Models work fine in the classification for reduced dimensions. High dimensionality cause problems since the amount of training data may become insufficient, or computation time increases too much [22]. One option is to reduce the number of features preprocessing the data, for example using the Principal component analysis (PCA) procedure. However, it would cause a loss of information for the classification, and would require extra data preprocessing.

We have decided to use as a Machine Learning algorithm the **Bayesian network** approach for several reasons. First of all, we are working with the users' behavioural patterns **statistical data**, and we assume the **conditional independence** among the variables. Furthermore, Bayesian networks allow us to perform inference on the network **without feeding all the variables**. Finally, there are a set of C++ libraries SMILE and SMILearn that allow us easily to use a Bayesian network generated graphically with ROS.

Artificial Neural Network can be perfectly used in our approach as well. If a future study suggested that the variables are not conditionally independent, a good alternative would be the use of ANN.

3.1.2 Parser

The parser package has been implemented as a ROS Service (`http://wiki.ros.org/Services`) due to the fact that it could be defined by a pair of messages. One message for the request: an absolute path to an ELAN file. And another message for the response: all the annotations parsed.

To parse ELAN annotation files we use TinyXML-2 [28] because it is a small, simple, operating system independent, free and open source XML parser for the C++ language.

Annotations

ELAN files contain annotations about behavioural features that are used to train and perform inference in the Bayesian network. Generally, there are thirteen different kind of annotations that are listed in Table 3.1. However, this module is prepared to parse all the annotations independently of the name and the quantity.

Server Specification

The `parse_data` service call request takes a string with the absolute path to an ELAN file, and returns the following data:

- **parse_data**
  - **data** is a vector of AnnotationLists:
    - **id** is a string with the AnnotationList identifier. For instance: `usr_mov, rob_mov, usr_gesture, usr_present, etc.`
    - **list** is a vector of Annotations:
      - **text** is a string with the annotation text
      - **tini** is an integer with the annotation initial time in milliseconds
      - **tend** is an integer with the annotation ending time in milliseconds

Tier annotation	Description
usr_sp	Anything the user says.
usr_cmd	Commands given by the user to the robot: `follow, stop, back, forward, turn_left, turn_right, turn_around.`
usr_present	Item that the user presents to the robot: `cupboard, table, cup, room, etc.`
usr_announce	When the user informs about what is (s)he going to show next. For instance: `"now we go to the office"`
usr_irrel_sp	Mumblings, fillers, etc.
usr_confirm	Any kind of user praise and confirmation.
usr_mov	Users movements, mainly adjustments to the robot. For example: `adjustment_further, adjustment_closer, adjustment_to_front, adjustment_left, guide, etc.`
rob_sp	Anything the robot says.
rob_prompt	Speech from the robot when it is ready to accept instructions: `hello, lost, move_on, show_me.`
rob_confirm	Any kind of robot confirmation: `break, ready, found, follow, stop_follow, done, etc.`
rob_mov	Robot movements: `forward, backwards, explore, follow, turn_left, etc.`
usr_gesture	Gestures performed by the user: `fingertip_point, hold_and_fingertip, touch_item_full_hand, hold_item, hand_point, sweep_wave.`
pause	Interruptions during the experiment by technical issues.

Table 3.1: Different kind of annotations in ELAN files

3.1.3 ELAN Translator

This package calls the `data_parser` to obtain all the annotations. Only 4 tier annotations are used to extract 5 different user's interactions patterns, and also 1 more tier is used to determine the classification of the item presented. Those annotations are the one that contain the most relevant interaction patterns:

- **usr_cmd** : Commands given to the robot.
- **usr_present**: The classification of the item being presented by the user. `Object, Workspace, Region` and sometimes `Unknown`.
- **usr_announce**: Boolean variable to indicate whether the user announces the item that is going to show next.
- **usr_mov**: From this annotation two different variables enter in the Bayesian network, Heading Adjustment and Distance Adjustment. When users change their angular position with respect to the robot they perform Heading Adjustment. If users

change their distance with respect to the robot then they perform Distance Adjustment.

- **usr_gesture**: Gesture performed by the user.

Those annotations are strings that are interpreted, transformed to Bayesian network variables, and published sequentially in 6 different topics (Figure 3.1). Furthermore, one more topic `ELAN_trigger` is used to determine when a user show-episode is finished. This topic will be useful in another nodes such as `interaction_recognition` to perform inference in the Bayesian Network.

The `elan_translator` is a publisher (`http://wiki.ros.org/Topics`) because once the user starts interacting with the robot, (s)he starts producing information. There is no request from another module, therefore the communication through topics is the most suitable one.

Publishers Specification

- **ELAN_dist_adj**
  - **value** is an integer variable of the Bayesian network. When the user changes his distance with respect to the robot it has value `1`, otherwise `0`.
  - **tini** is an integer with the annotation initial time in milliseconds
  - **tend** is an integer with the annotation ending time in milliseconds
- **ELAN_category**
  - **value** is an integer variable of the Bayesian network with the item category when an item is being presented. It has the following values: `object=0`, `region=1`, `workspace=2`, `unknown=3`.
  - **tini** is an integer with the annotation initial time in milliseconds
  - **tend** is an integer with the annotation ending time in milliseconds
- **ELAN_heading_adj**
  - **value** is an integer variable of the Bayesian network. When the user performs a heading adjustment with respect to the robot it has value `1`, otherwise `0`.
  - **tini** is an integer with the annotation initial time in milliseconds
  - **tend** is an integer with the annotation ending time in milliseconds
- **ELAN_trigger**
  - **data** Boolean variable that is published as `true` whether the user finishes presenting an item. When there are no more annotations to publish it is indicated publishing `false`.

- **ELAN_last_cmd**
  - **value** is an integer variable of the Bayesian network that has the following values depending on the command given to the robot: `back=0`, `follow=1`, `forward=2`, `stop=3`, `turn=4`, `none=5`.
  - **tini** is an integer with the annotation initial time in milliseconds
  - **tend** is an integer with the annotation ending time in milliseconds
- **ELAN_gesture**
  - **value** is an integer variable of the Bayesian network that has the following values depending on the gesture performed by the user. If the user points with the finger: `fingertip_point=0`, or if (s)he points with the full hand: `hand_point=1`. If the user grabs and holds the object: `hold_item=2`, in the case that the gesture is a sweep with the hand: `sweep_wave=3`. Also the user can touch the item without holding it: `touch_full_hand=4` or if the subject does not perform any gesture: `none=5`.
  - **tini** is an integer with the annotation initial time in milliseconds
  - **tend** is an integer with the annotation ending time in milliseconds
- **ELAN_announce**
  - **value** is an integer variable of the Bayesian network. When the user announces the item that is going to show next it has value `1`, otherwise `0`.
  - **tini** is an integer with the annotation initial time in milliseconds
  - **tend** is an integer with the annotation ending time in milliseconds

3.1.4 Interaction Monitor

The `interaction_monitor` node contains a short memory module to store all the user's behavioural features that occur around every object, workspace, region presentation.

This package is subscribed to 7 different topics, where 6 topics are behavioural features that will feed the Bayesian network. Those topics come from the ELAN Translator module explained in section 3.1.3. However, this prototype has been designed considering future contributions, therefore all those behavioural features can be obtained from other sources. For instance, a future work that probably is coming soon is the user gesture recognition using a Kinect.

The seventh topic that this node is subscribed to is a trigger that will generate the publication of the `BN_vars` topic. This trigger topic is also used to indicate that there are no more behavioural features coming. Then, the `interaction_learner` module will generate a Bayesian network, or the `interaction_recognition` will print some statistics evaluating the performance.

The `BN_vars` topic (Figure 3.1) represents the last behavioural features made by the user with no more than 10 seconds before an item presentation.

In this case, the interaction_monitor is also a publisher because once the user starts interacting with the robot, (s)he starts producing information. There is no request from another module, therefore the communication through topics is the most suitable one.

Publisher Specification

- **BN_vars**
  - **last_cmd:** Integer that codifies the last command given to the robot before the presentation of an item.
  - **announce:** Integer that codifies if the user announced the item presented.
  - **gesture:** Integer that codifies the gesture performed by the user when presenting an item.
  - **head_adj:** Integer that codifies if the user changed his angular position with respect to the robot.
  - **dist_adj:** Integer that codifies if the user changed his distance with respect to the robot
  - **category:** Integer that codifies the category of the object presented.

3.1.5 Interaction Learner

This package is used to train a Bayesian network, it is subscribed to the /BN_vars topic published by the interaction_monitor node (Figure 3.1) and it generates a file with a Bayesian network.

We are using the decision-theoretic models tools developed by the Decision Systems Laboratory at the University of Pittsburgh [2]:

- **GeNIe** is a graphical environment for building graphical decision-theoretic models like Bayesian networks.

- **SMILE** is a platform independent library of C++ classes for reasoning in graphical probabilistic models, such as Bayesian networks and influence diagrams

- **SMILearn** is a specialized module that extends functionality provided by SMILE by providing a set of classes that implement learning algorithms and other useful tools for automated building graphical models from data.

Firstly, this node opens a GeNIe/SMILE file that contains the structure of a Bayesian network. This Bayesian network structure (Figure 3.2) has been designed with GeNIe, and it has six different variables previously mentioned. Five of them are different interaction patterns, and the other one is the item category presented by the user.

After that, this node subscribes to the /BN_vars topic and collects all the messages. Once the node has received all the annotations, it trains the Bayesian network and generates a new GeNIe/SMILE file using SMILearn library.

Those Bayesian networks that the package opens and generates are saved in a folder of the Interaction Recognition package.

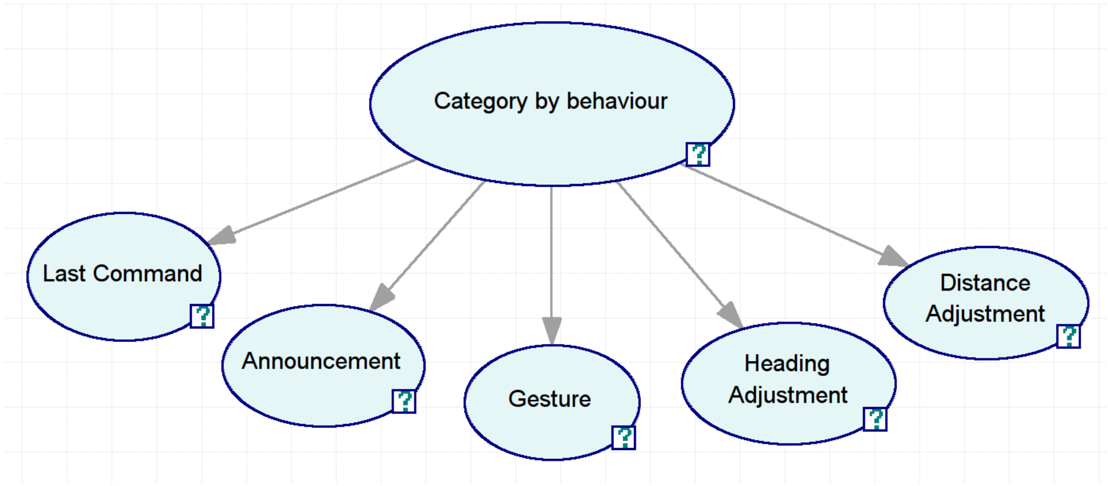

Figure 3.2: GeNIe Bayesian network

3.1.6 Interaction Recognition

With the help of the SMILE library this node performs inference in the Bayesian network generated by the `interaction_learner` package.

First of all, the `interaction_recognition` node opens a GeNIe/SMILE file with a trained Bayesian Network (Figure 3.2). Then, this node subscribes to the `/BN_vars` topic and for each message that arrives performs inference in the network to obtain the `Category by behaviour` posterior probabilities.

For each classification (`Object, Workspace, Region` and `Unknown`) the posterior probability is computed and printed. If the highest posterior probability has a difference less than a 10% with the second highest one, we consider that they are too similar and there is an ambiguity. For example, in figure 3.3 region and workspace categories have the higher posterior probability values with a difference less than 10%.

```
P("category" = object) = 0.009547
P("category" = region) = 0.420887
P("category" = workspace) = 0.495601
P("category" = unknown) = 0.073965
```

Figure 3.3: Output of the node when we perform inference in the Bayesian network. Region and Workspace posterior probabilities are very similar.

The `usr_present` annotation is not used to perform inference in the network when we are recognising the `Category by behaviour`. However, this annotation is compared among the highest posterior probabilities to check the performance of the network.

Output statistics

When no more annotations are being published, a statistical overview is printed to check the results of the Bayesian network. The table 3.2 contains the description of the statistical data.

Output	Description
Matches	Total number of matches. It is a match if the highest posterior probability has at least a 10% of difference between the second highest, and it coincides with the `usr_present` annotation
Mismatches	Total number of mismatches. It is a mismatch if the highest posterior probability has at least a 10% of difference between the second highest, and it doesn't coincide with the `usr_present` annotation
Similar between 2	When the two highest posterior probability values have a difference less than a 10%, and one of them coincides with the `usr_present` annotation.
Similar among 3	When the three highest posterior probability values have a difference less than a 10%, and one of them coincides with the `usr_present` annotation.
All are Similar	When all of them have a posterior probability difference less than a 10%
Unknown category classified	Number of items that were defined as `Unknown` and have been classified as `Object, Workspace` or `Region`.
Similar between 2, mismatch	When the two highest posterior probability values have a difference less than a 10%, and no one coincides with the `usr_present` annotation.
Similar among 3, mismatch	When the three highest posterior probability values have a difference less than a 10%, and no one coincides with the `usr_present` annotation.

Table 3.2: Meaning of the statistics

3.2 User movements from the Tracker

This module is the first extension of the Interaction Patterns Manager in which we try to extract behavioural features automatically without the preprocessed ELAN annotations files. The aim of this part is to check whether or not we can automate the extraction of those behavioural features related to the user movement detected by the tracker, `Heading_adjustment` and `Distance_adjustment` (Figure 3.4).

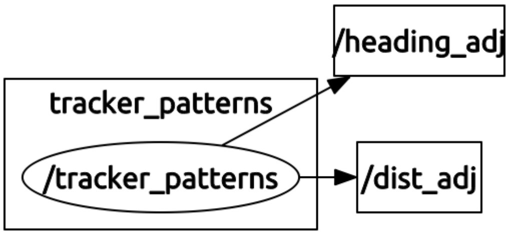

Figure 3.4: User movements from the Tracker prototype nodes and topics structure

This module has been implemented considering as well all the data recorded in the study case "Understanding Spatial Concepts from User Actions" [31]. Concretely, we extract the users' Interaction Patterns from the trajectories generated by a tracker using the laser scanner and checking the odometry readings.

3.2.1 Tracker Patterns

First of all, the node opens a trajectory file and starts checking the subject movement. If the user has moved more than an established threshold in meters, we are in a possible case of `Distance_adjustment`. If the subject has changed his angular position relative to the robot more than an established threshold in radians, we are in a possible case of `Heading_adjustment`.

Once a possible case of `Distance_adjustment` or `Heading_adjustment` is detected, the next step is to check if those movements were produced by the movement of the robot in the odometry file. If the robot was not moving, the node will consider that the user performed an adjustment movement and will publish that information in the corresponding topic.

Publisher Specification

- **heading_adj**
  - **value** is an integer variable of the Bayesian network. When the user performs a heading adjustment to the robot it has value `1`, otherwise `0`.
  - **tini** is an integer with the current time in milliseconds
  - **tend** is an integer with the current time in milliseconds

- **dist_adj**
  - **value** is an integer variable of the Bayesian network. When the user performs a distance adjustment to the robot it has value `1`, otherwise `0`.
  - **tini** is an integer with the current time in milliseconds
  - **tend** is an integer with the current time in milliseconds

Chapter 4

Evaluation

37 different subjects participated in the study case "Understanding Spatial Concepts from User Actions" [31] guiding and interacting with a robot, where they presented several objects, workspaces, and regions in a part of the Computer Science department at the Lund University.

That means that we have 37 different ELAN annotation files with 548 presentations and their corresponding interaction patterns to test the **Interaction Patterns Manager** prototype. Furthermore, we also have the laser scanner and odometer recordings that will be used to test the **User movements from the Tracker** prototype.

4.1 Interaction Patterns Manager

We have done several tests with all of the subjects in the dataset. We have considered in all the tests to mix different subjects from the beginning and the end of the study, in order to avoid that the exhaustion during the recording and the preprocessing data in the previous study can affect our results.

Test 1 is based on all-against-all, the Bayesian network is trained using all the dataset and we perform inference as well with all the dataset. Test 2 and 3 try to divide the dataset in two halves. In the last test 18.5% of the dataset are used to train and 81.5% to perform inference in the Bayesian network.

4.1.1 Test 1

In this case we are using all the dataset to train the Bayesian network, and the same data to perform inference. More detailed information about the used datasets can be found in table 4.1.

Figure 4.1 shows the results after performing inference with all the dataset. A total of 226 presentations **matched**. The algorithm has been able to classify 40 of the presentations

Training set	
Number of files	37
Number of presentations	548
Subject files	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37
Test set	
Number of files	37
Number of presentations	548
Subject files	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37

Table 4.1: Datasets in test 1

that were classified previously as **unknown**. And then, we have two big categories such as **mismatches** and **similar between two** that are of great interest to be analysed in more detail.

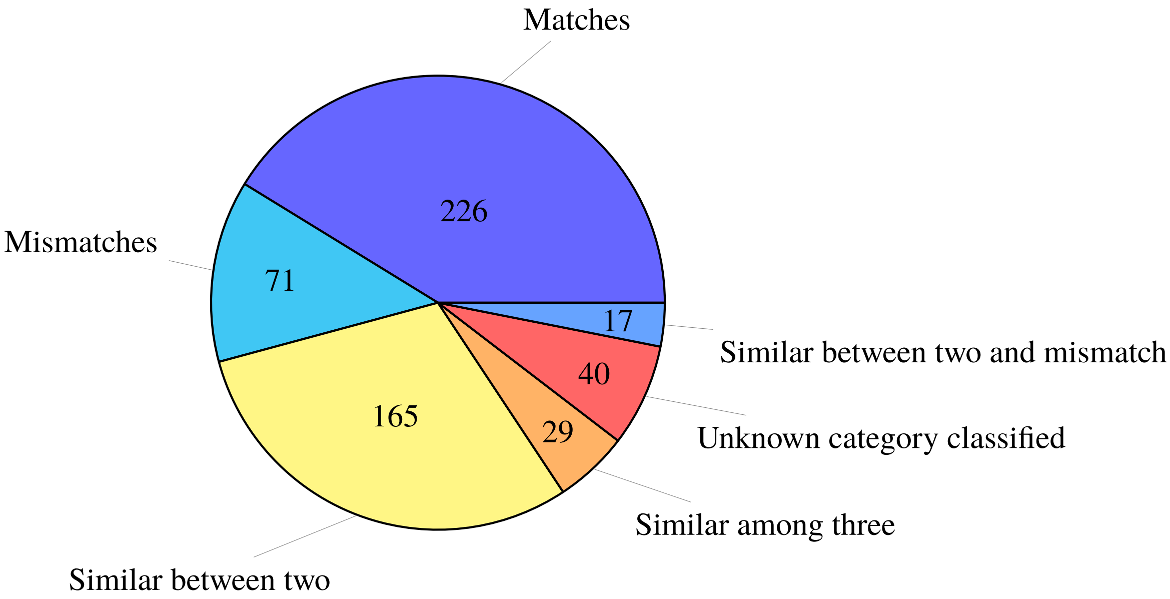

Figure 4.1: Results of test 1

Analysing more deeply the results of the two biggest groups that are not **matches** we can see that the results are not bad at all, in the case of mismatches we have got the following:

- **Mismatches** 71
  - 40 objects are classified as a workspace. Mainly the object "chair" is classified as a workspace. Other objects misclassified are "phone", "dustbin", "printer" and "paper".

- 17 workspaces are classified as a region. In this case, "printer" is one of the most misclassified followed by "microwave", "table".
- 6 regions are classified as a workspace. "meeting room" is the most misclassified.
- 4 workspaces are classified as an object. Those are "printer" and "fridge"
- 2 workspaces and 1 region are classified as unknown.
- 1 object is classified as a region, "dustbin".

What is a chair? Is it really an object or a workspace? We have realised during this study that there are a lot of objects/workspaces that we cannot really decide which is the most suitable category. Furthermore, the same happens with workspace/region category.

If we analyse the results when the algorithm cannot decide just one category but has two possible options and one of them is the correct category, we can see that the results are quite good as well:

- **Similar between two** 165
  - 95 presentations have similar results between object and workspace
  - 58 presentations have similar results between region and workspace
  - 12 presentations have similar results between any category and unknown

As we have mentioned before, some objects have an unclear category. Normally, those categories are between objects and workspaces, and regions and workspaces. We can consider that we have good results when we don't have any case where the algorithm is not able to distinguish between region and object.

4.1.2 Test 2

19 ELAN files corresponding to subjects tagged with an even number are used to train the Bayesian network, and the rest of the dataset to test the results. More detailed information about the used datasets can be found in table 4.2.

Training set	
Number of files	19
Number of presentations	277
Subject files	1, 3, 5, 7, 9, 11, 13, 15, 17 19, 21, 23, 25, 27, 29, 31, 33, 35, 37

Test set	
Number of files	18
Number of presentations	271
Subject files	2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36

Table 4.2: Datasets in test 2

Figure 4.2 shows the results after performing inference with the other half of the dataset. A total of 111 presentations **matched**. The algorithm has been able to classify 21 of the presentations that were classified previously as **unknown**. As in the previous test there are two big categories **mismatches** and **similar between two** that are of great interest to be analysed in more detail.

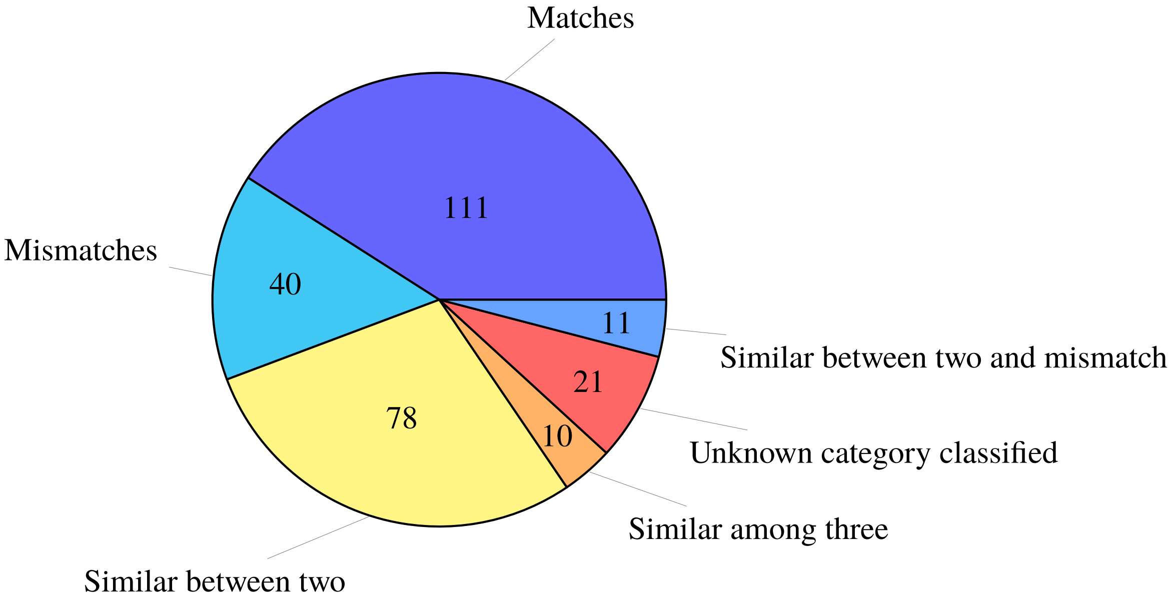

Figure 4.2: Results of test 2

Analysing more deeply the results of the two biggest groups that are not **matches** we can see that the results are not bad at all, in the case of mismatches we have the following results:

- **Mismatches** 40
  - 20 objects are classified as a workspace. Mainly the object "chair" is misclassified. Other objects misclassified are "laptop", "dustbin" and "lamp".
  - 9 workspaces are classified as a region. In this case, "printer room" is one of the most misclassified followed by "microwave" and "table".
  - 7 regions are classified as a workspace. "meeting room" and "office" are the most misclassified.
  - 1 workspace is classified as object.
  - 2 workspaces and 1 object are classified as unknown.

If we analyse the results when we have two candidate categories and one of them is the correct one, we can see that the results are quite good as well:

- **Similar between two** 78
  - 52 presentations have similar results between object and workspace
  - 17 presentations have similar results between region and workspace
  - 9 presentations have similar results between any category and unknown

4.1.3 Test 3

18 ELAN files are used to train the Bayesian network. Those subjects tagged with an odd number are used to train, and the 19 other subjects to test the results. More detailed information about the used datasets can be found in table 4.3.

Training set	
Number of files	18
Number of presentations	271
Subject files	2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36
Test set	
Number of files	19
Number of presentations	277
Subject files	1, 3, 5, 7, 9, 11, 13, 15, 17 19, 21, 23, 25, 27, 29, 31, 33, 35, 37

Table 4.3: Datasets in test 3

Figure 4.3 shows the results after performing inference with the other half of the dataset. A total of 116 presentations **matched**. The algorithm has been able to classify 23 of the presentations that were classified previously as **unknown**. As in the previous tests there are two big categories **mismatches** and **similar between two** that are of great interest to be analysed in more detail.

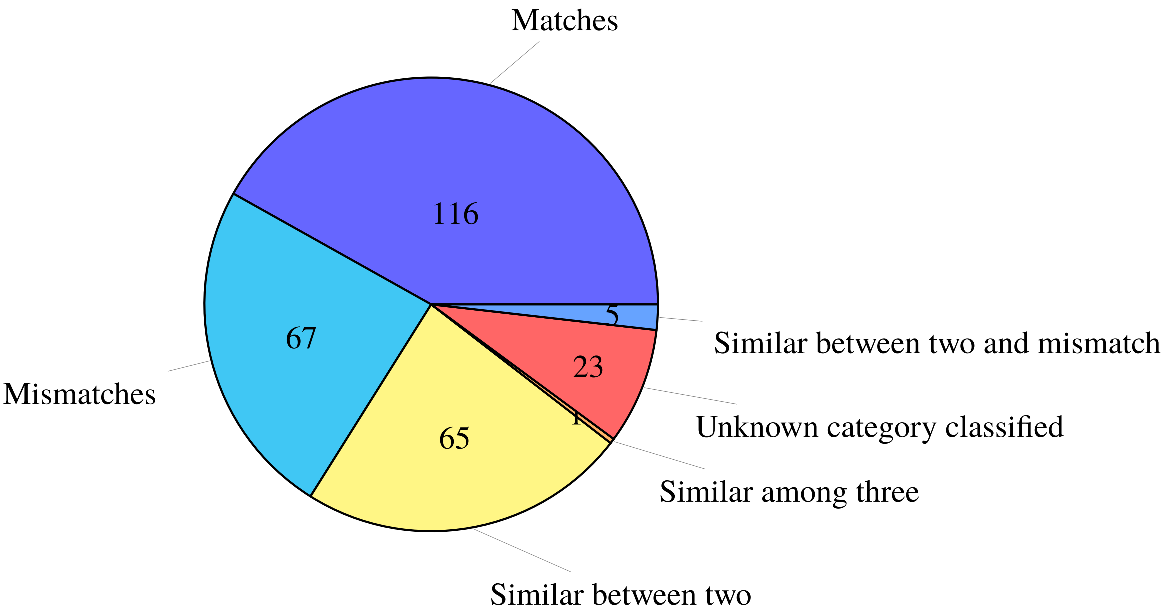

Figure 4.3: Results of test 3

Analysing more deeply the results of the two biggest groups that are not **matches** we can see that the results are not bad at all, in the case of mismatches we have the following results:

- **Mismatches** 67
  - 23 objects are classified as a workspace. In this case we have a variety of objects: "chair", "dustbin", "paper", "stapler" and "phone".
  - 16 regions are classified as a workspace. "meeting room", "office" and "printer room" are the most misclassified.
  - 16 workspaces are classified as objects. There is a variety of workspaces such as: "projector", "table", "microwave", "shelf".
  - 11 workspaces are classified as a region. In this case, "printer room" is one of the most misclassified.
  - 1 object is classified as a region, "dustbin".

If we analyse the results when we have two candidate categories and one of them is the correct one, we can see that the results are quite good:

- **Similar between two** 65
  - 39 presentations have similar results between object and workspace
  - 20 presentations have similar results between region and workspace
  - 6 presentations have similar results between any category and unknown

4.1.4 Test 4

7 subjects are used to train the Bayesian network. Concretely, those ELAN files with a multiple of 5 will be used to train the network, and this corresponds about the 18.5% of the presentations. The rest of the dataset is used to test the network. More detailed information about the used datasets can be found in table 4.4.

Training set	
Number of files	7
Number of presentations	101
Subject files	5, 10, 15, 20, 25, 30, 35

Test set	
Number of files	30
Number of presentations	447
Subject files	1, 2, 3, 4, 6, 7, 8, 9, 11, 12, 13, 14, 16, 17, 18, 19, 21, 22, 23, 24, 26, 27, 28, 29, 31, 32, 33, 34, 36, 37

Table 4.4: Datasets in test 4

Figure 4.4 shows the results after performing inference with the 81.5% remaining dataset. A total of 167 presentations **matched**. 28 of the presentations were classified previously as **unknown**, and the algorithm has been able to classify. As in the previous

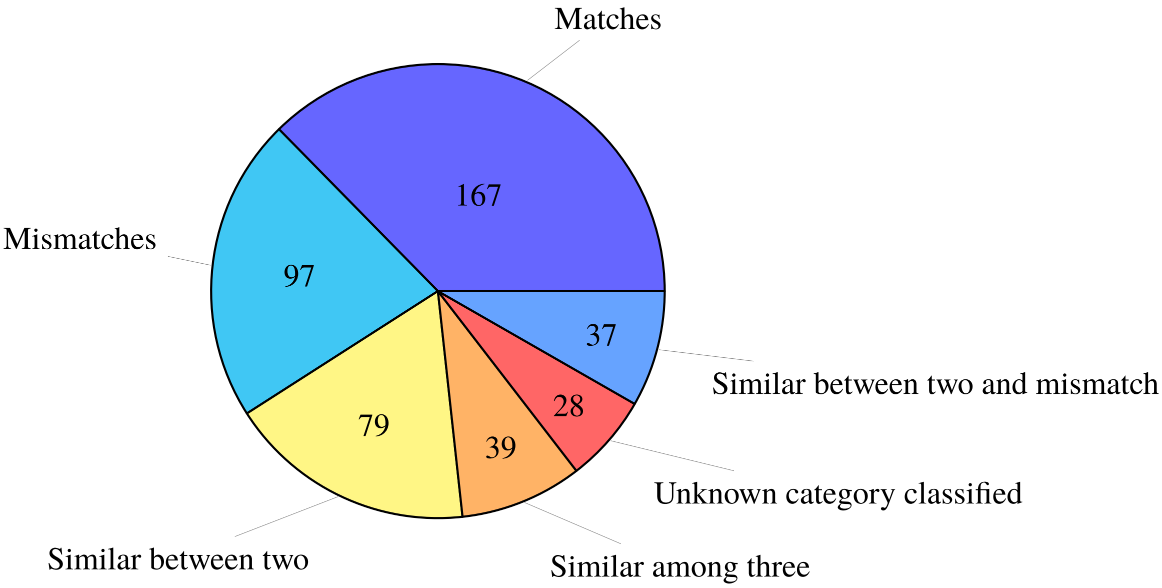

Figure 4.4: Results of test 4

tests there are two big categories **mismatches** and **similar between two** that are of great interest to be analysed in more detail.

Analysing more deeply the results of the two biggest groups that are not **matches** we can see that the results are not bad at all, in the case of mismatches we have the following results:

- **Mismatches** 97
  - 29 objects are classified as a workspace. The most misclassified object is "chair", followed by "paper".
  - 31 presentations were classified as unknown: 19 workspaces, 10 objects and 2 regions.
  - 15 regions are classified as a workspace. The most misclassified are "meeting room" and "office".
  - 9 workspaces are classified as a region. In this case, "printer room" "lucas entrance" are the most misclassified.
  - 7 workspaces are classified as objects, "projector", "photocopy machine" and "shelf"
  - 5 regions are classified as an object, "office".
  - 1 object is classified as a region.

If we analyse the results when we have two candidate categories and one of them is the correct one, we can see that the results are quite good:

- **Similar between two** 79
  - 33 presentations have similar results between object and workspace
  - 29 presentations have similar results between region and workspace
  - 17 presentations have similar results between any category and unknown

4.2 User movements from the Tracker

Even though we have laser scanner and odometry recordings of all the 37 subjects that participated in the previous study case, we only have the preprocessed tracker trajectories of two subjects (subject 10 and subject 21) to test this part of the prototype.

The evaluation has been done comparing the behavioural features extracted based on objective data (laser scanner and tracker), to the ELAN annotations that also contain behavioural features but were generated in a manually analysis effort.

This part of the prototype has been tested with the same thresholds for both subjects. The threshold to detect `distance adjustment` is established in 500mm during the last second. In the case of `heading adjustment` it is established in 45° as well during the last second.

4.2.1 Subject 10

According to the ELAN annotation file for subject 10, he did a total of 5 `heading adjustment` and 21 `distance adjustment` with respect to the robot during all the presentations in the guided tour.

On the other hand, the prototype has detected reading the trajectory and odometry files 7 `distance adjustments` and 7 `heading adjustments`. The results more in detail can be found in the table 4.5.

Analysing more in detail the results obtained with the videos, trajectory and odometry files we can extract the following:

- Four annotations 4, 9, 12, 28 have matched with the ELAN annotations.
- Six annotations 11, 13, 14, 25, 26, 27 are detected by the tracker, but in the ELAN file are considered as "guide". It means that the user was guiding the robot. Subjectively, all his movements are not considered adjustments, just guiding.
- Eleven annotations 1, 2, 3, 8, 15, 23, 24, 29, 30, 31, 32 are discarded because they were performed while the robot was moving. We discard annotations when the robot is moving to distinguish whether a distance or heading adjustment is performed by the user, and not by the robot.
- Six annotations 5, 6, 7, 10, 16, 17 are not detected because the subject performs a very small movement, less than 500mm. We have set in the prototype two thresholds to detect adjustment movements in an objective way. This is not the case of the ELAN annotations, where subjectively even a small movement can be considered has an adjustment.
- Five annotations 18, 19, 20, 21, 22 the robot was in a cluttered area (LUCAS room). Usually the tracker has troubles in those cases and cannot track correctly the subject.

The first column of the table 4.5 is the row **identifier**. The second and the third column indicate the **initial time** of the behavioural feature in the video, and the **final time**. Two columns indicate if in the ELAN file is considered any adjustment annotation: **ELAN**

distance adjustment and **ELAN heading adjustment**. The last two columns indicate if the algorithm that uses the information from the **tracker** detects some distance or heading adjustment.

id	**Time init**	**Time end**	**ELAN distance adjustment**	**ELAN heading adjustment**	**Tracker distance**	**Tracker heading**
1	00:13	00:15	adjustment closer	No	Yes	NO
2	00:22	00:23	No	adjustment front	No	Yes
3	00:37	00:39	adjustment closer	No	No	No
4	02:42	02:48	adjustment closer further	No	Yes	No
5	03:26	03:28	adjustment closer	No	No	No
6	03:32	03:35	adjustment further	No	No	No
7	04:46	04:47	adjustment closer	No	No	No
8	04:53	04:58	No	adjustment to front	No	No
9	05:25	05:31	adjustment closer	No	Yes	No
10	05:37	05:40	No	adjustment front	No	No
11	05:45	05:45	No	No	Yes	No
12	05:53	05:57	adjustment closer	No	Yes	No
13	06:10	06:10	No	No	Yes	Yes
14	06:22	06:22	No	No	No	Yes
15	06:52	06:56	adjustment closer	No	No	No
16	07:17	07:19	adjustment further	No	No	No
17	07:21	07:24	adjustment further	No	No	No
18	07:40	07:43	adjustment closer	No	No	No
19	07:44	07:47	adjustment further	No	No	No
20	07:53	08:00	adjustment closer further	No	No	No
21	08:05	08:08	adjustment closer	No	No	No
22	08:09	08:12	adjustment further	No	No	No
23	09:45	09:51	No	adjustment to front	No	No
24	10:05	10:09	adjustment closer	No	No	No
25	10:12	10:12	No	No	No	Yes
26	10:18	10:18	No	No	No	Yes
27	10:23	10:23	No	No	No	Yes
28	10:36	10:40	adjustment closer further	No	Yes	Yes
29	10:41	10:42	adjustment closer	No	No	No
30	10:45	10:47	adjustment closer	No	No	No
31	12:20	12:27	No	adjustment to front	No	No
32	12:28	12:30	adjustment closer further	No	No	No

Table 4.5: Heading adjustment and Distance adjustment summary for subject 10

Figure 4.5 shows an overview of the results obtained.

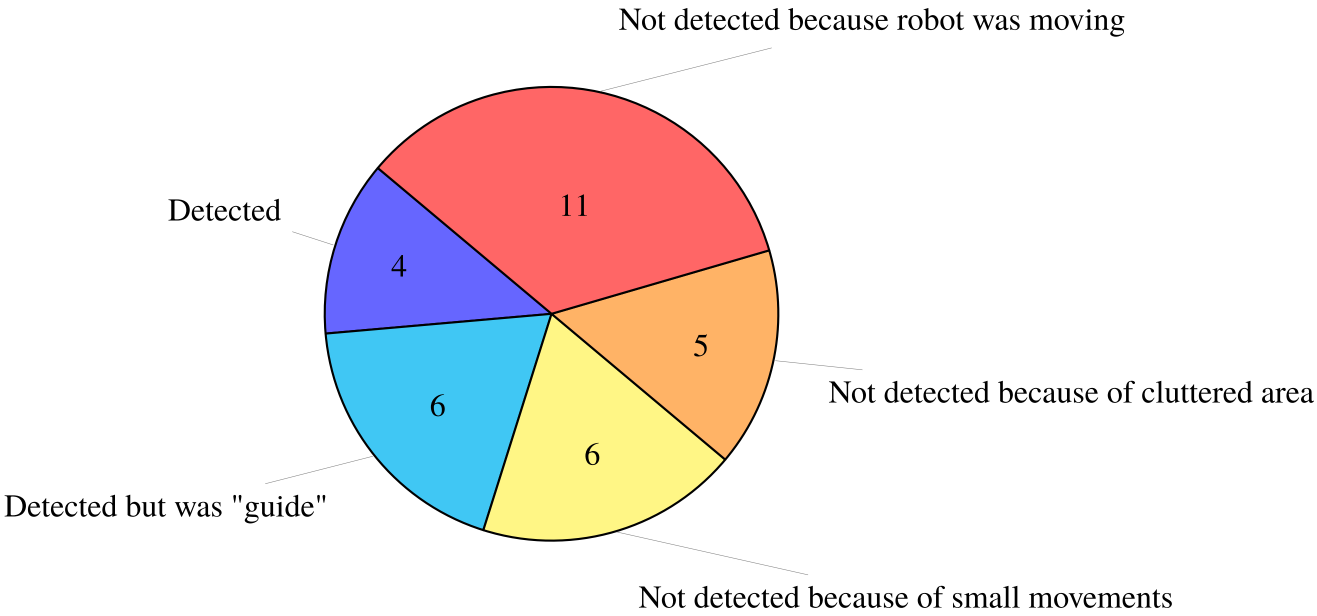

Figure 4.5: Results after testing subject 10

4.2.2 Subject 21

According to the ELAN annotation file for subject 21, she did a total of 11 `heading adjustment` and 8 `distance adjustment` to the robot during all the presentations in the guided tour.

On the other hand, the prototype has detected reading the trajectory and odometry files 12 `distance adjustments` and 9 `heading adjustments`. The results more in detail can be found in the table 4.6.

Analysing more in detail the results obtained with the videos, trajectory and odometry files we can extract the following:

- Eight annotations 2, 5, 8, 12, 14, 17, 18, 19 have a very good result. It is important to mention that in some cases (for example annotation number 2) only was considered the annotation "ELAN heading adjustment". However, the adjustment was from left to front and it also involves some distance adjustment.

- Annotation 6 was not in the ELAN file, but the user really performs the adjustment.

- Four annotations 4, 7, 10, 11 are detected by the tracker but in the ELAN file are considered as "guide". It means that the user was guiding the robot. Subjectively, all his movements are not considered adjustments, just guiding.

- Annotation number 13 is discarded because it was performed while the robot was moving. We discard annotations when the robot is moving to distinguish whether a distance or heading adjustment is performed by the user, and not by the robot.

- Three annotations 1, 3, 15 are not detected because the subject performs a very small movement, or just moves the upper part of the body. The laser scanner was located in the robot, below the knee height. The tracker uses the laser data to detect the legs of the user. Therefore, movements done just with the upper part of the body are not detected. Furthermore, if the movement performed by the user doesn't reach any threshold, the algorithm doesn't consider the movement as an adjustment.
- Two annotation 9 and 16 are not detected.
- Three annotations 20, 21, 22 the robot was in a cluttered area (LUCAS room). Usually the tracker has troubles in those cases and cannot track correctly the subject.

id	Time init	Time end	ELAN distance adjustment	ELAN heading adjustment	Tracker distance	Tracker heading
1	01:04	01:06	No	adjustment left to front	No	No
2	01:08	01:09	No	adjustment left to front	Yes	Yes
3	01:17	01:20	No	adjustment front to left	No	No
4	01:35	01:36	No (guide)	No (guide)	Yes	Yes
5	03:38	03:47	No	adjustment front	Yes	No
6	04:32	04:33	No	No	Yes	Yes
7	05:59	06:09	No (guide)	No (guide)	Yes	Yes
8	06:17	06:22	adjustment closer	No	Yes	No
9	06:40	06:44	No	adjustment back front	No	No
10	06:47	06:47	No (guide)	No (guide)	No	Yes
11	06:53	06:53	No (guide)	No (guide)	No	Yes
12	06:55	07:01	No	adjustment right to front	Yes	Yes
13	07:08	07:11	No	adjustment front to right	No	No
14	07:14	07:23	No	adjustment right to front	Yes	Yes
15	07:23	07:31	adjustment further closer	No	Yes	Yes
16	08:09	08:15	adjustment closer	No	No	No
17	11:08	11:14	adjustment closer further	No	Yes	No
18	11:18	11:23	adjustment closer/further	No	Yes	No
19	12:45	12:48	adjustment further	No	Yes	No
20	12:53	12:55	adjustment closer	No	No	No
21	13:57	13:59	No	adjustment front	No	No
22	14:12	14:15	No	adjustment front	No	No

Table 4.6: Heading adjustment and Distance adjustment summary for subject 21

Figure 4.6 shows an overview of the results obtained.

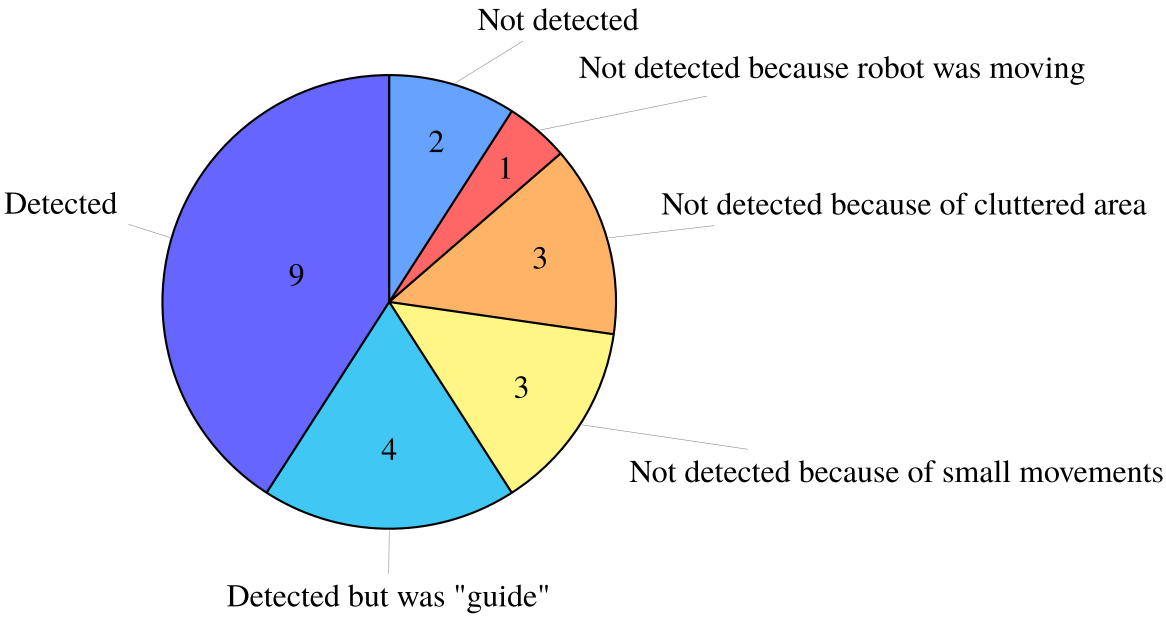

Figure 4.6: Results after testing subject 21

Chapter 5

Discussion and Conclusions

In this chapter we present the conclusions about our approach and the prototype implementation. Furthermore, we also introduce some ideas about future research.

5.1 Interaction Patterns Manager

The results obtained from the implementation of the first prototype, where we use behavioural features from annotation files to recognise the presentation category, are really good and suggest that interaction patterns can be used to help in the understanding and recognition of users' intentions.

All those mismatching cases, normally when deciding between object/workspace or workspace/region, should be studied in more detail. Maybe adding a new category such as "big object" could help in obtaining better results.

The small but obvious difference in the number of mismatches between test 2 and test 3 indicates that some subjects have a behavioural pattern more generic than the other subjects. It is important to consider several subjects in the learning process that can generalise the trend instead of memorise a specific behaviour.

The results obtained in the test 4, in which only 18.5% of the dataset have been used in the learning process, show that even though we use a small dataset to train the network, it does not compromise the learning process.

5.2 User movements from the Tracker

We tested the second module with only two different subjects, and the results obtained are quite different. Considering only the figures, subject 21 has much better results than subject 10. However, those results contain a multitude of considerations.

One of the important things that we have to bear in mind is that our prototype does not consider any distance or heading adjustment when the robot is moving. The main reason of this is because it is much easier to distinguish when a distance or heading adjustment is performed by the user, and not by the robot. It does not affect our approach that much because most of the presentations are done when the robot is not moving.

Annotations from ELAN files are very subjective, even a small movement can be considered as an adjustment. That is not the case of the prototype "User movements from the Tracker." Our prototype is very rigid and has only two thresholds to tune the algorithm: Distance and Angle. During the tests the distance threshold was established to 500mm and the Angle threshold to 45°.

Our prototype continuously processes if the subject has done a distance or a heading adjustment, it is not prepared to distinguish when the user is guiding the robot. That is why in both tests both subjects have a considerable number of adjustments detected when they were guiding the robot. However, this issue will not affect the connection of both prototypes, because normally the presentations are not done immediately after the guiding process. Furthermore, some adjustment patterns are usually performed after guiding the robot and before the presentation.

The tracker that we were using to calculate the users' trajectories had some troubles in cluttered environments such as rooms full of chairs. Although we could not test our approach properly in those cases, we consider that the results would not change too much. With an improvement of the tracker those problems should disappear.

As a final conclusion of this second prototype, we can perfectly use the tracker to obtain heading and distance adjustment. However, those interaction patterns will be different than the ones obtained in a manual effort. It means that we have to test again the first prototype with this module integrated in order to evaluate this part correctly.

5.3 Future Work

The "Interaction Patterns Manager" prototype only uses ELAN annotation files to learn the users' behaviour and to recognise the users' intentions. Therefore, a future work could be the designing and implementation of modules to automatically extract behavioural features in order to use the whole system in live.

The first module implemented to automatically extract some behavioural features has been the "User movements from the tracker". This module is still not connected, therefore is of great interest this step of integration to perform more tests.

More devices can be added to the service robot such as a kinect camera for gesture recognition, or a microphone for speech recognition. Figure 5.1 shows a proposed approach for the extension and connection of futures modules. After the implementation of the proposed approach several tests could be done to compare the results with the one obtained using the annotation files.

We have used the Bayesian network technique as a machine learning algorithm in the "Interaction Patterns Manager". However, there are a lot of machine learning techniques in the literature that can be used as well. It could be very interesting to test different algorithms and compare the results. Fortunately, only one single module contains the Bayesian network algorithm, and it would be very easy to exchange with another module.

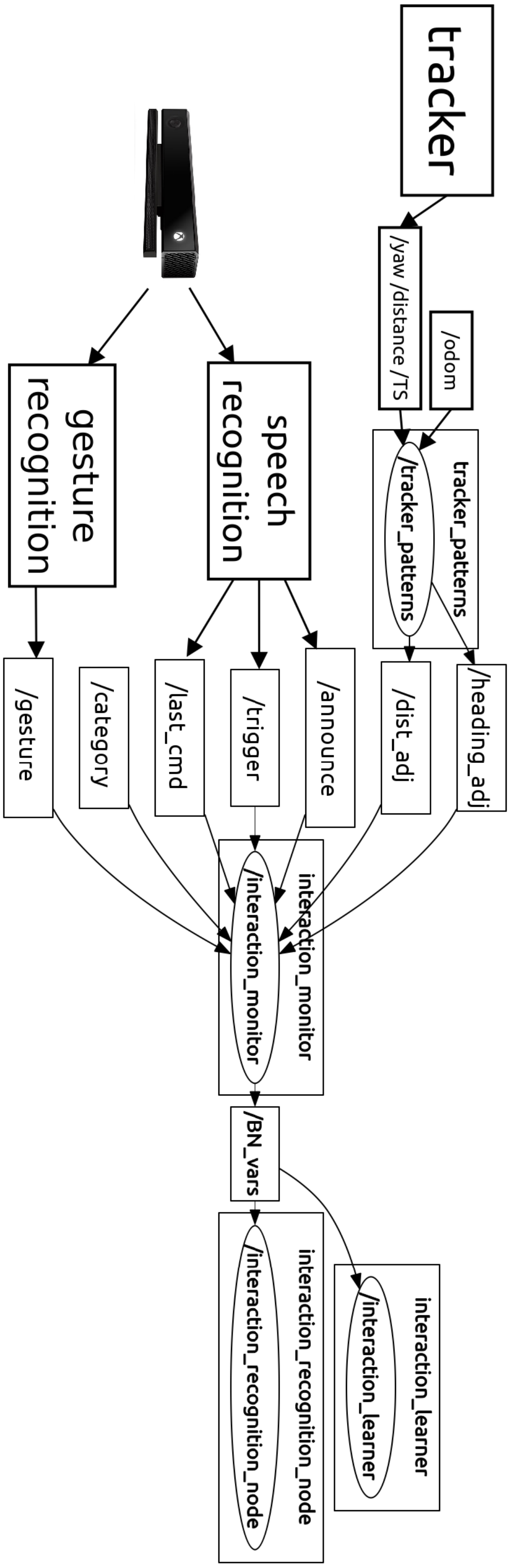

Figure 5.1: Future extensions

Bibliography

[1] The Language Archive. ELAN Multimedia Annotator. `https://tla.mpi.nl/tools/tla-tools/elan`.

[2] Decision Systems Laboratory at University of Pittsburgh. GeNIe & SMILE. `https://dslpitt.org/genie/`.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[4] Rebecca J. Brand, Dare A. Baldwin, and Leslie A. Ashburn. Evidence for 'motionese': modifications in mothers' infant-directed action. *Developmental Science*, 5(1):72–83, 2002.

[5] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[6] Anne Fernald and Claudia Mazzie. Prosody and focus in speech to infants and adults. *Developmental Psychology*, page 209221, 1991.

[7] Open Source Robotics Foundation. Robot Operating System (ROS). `http://www.ros.org/`.

[8] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[9] Dylan F. Glas, Phoebe Liu, Takayuki Kanda, and Hiroshi Ishiguro. Can a social robot train itself just by observing human interactions? In *Workshop on Machine Learning for Social Robotics, ICRA*, 2015.

[10] A. Haasch, N. Hofemann, J. Fritsch, and G. Sagerer. A multi-modal object attention system for a mobile robot. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 2712–2717, Aug 2005.

[11] H. Huettenrauch, K. Severinson Eklundh, A. Green, and E.A. Topp. Investigating spatial relationships in human-robot interaction. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5052–5059, Oct 2006.

[12] Jungsik Hwang, KunChang Lee, and Jaeyeol Jeong. A bayesian network approach to investigating user-robot personality matching. In Runhe Huang, AliA. Ghorbani, Gabriella Pasi, Takahira Yamaguchi, NeilY. Yen, and Beijing Jin, editors, *Active Media Technology*, volume 7669 of *Lecture Notes in Computer Science*, pages 124–133. Springer Berlin Heidelberg, 2012.

[13] Jana M. Iverson, Olga Capirci, Emiddia Longobardi, and M. Cristina Caselli. Gesturing in mother-child interactions. *Cognitive Development*, 14(1):57 – 75, 1999.

[14] T. Kanda, H. Ishiguro, M. Imai, and T. Ono. Development and evaluation of interactive humanoid robots. *Proceedings of the IEEE*, 92(11):1839–1850, Nov 2004.

[15] Thomas Kollar, Anu Vedantham, Corey Sobel, Cory Chang, Vittorio Perera, and Manuela Veloso. A multi-modal approach for natural human-robot interaction. In *Social Robotics*, pages 458–467. Springer, 2012.

[16] C. Lee and Yangsheng Xu. Online, interactive learning of gestures for human/robot interfaces. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 4, pages 2982–2987 vol.4, Apr 1996.

[17] J.F. Maas, T. Spexard, J. Fritsch, B. Wrede, and G. Sagerer. Biron, what's the topic? a multi-modal topic tracker for improved human-robot interaction. In *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, pages 26–32, Sept 2006.

[18] Michael G Madden. On the classification performance of TAN and general bayesian networks. *Knowledge-Based Systems*, 22(7):489–495, 2009.

[19] Ross Mead and Maja J Matarić. Toward robot adaptation of human speech and gesture parameters in a unified framework of proxemics and multimodal communication. In *Workshop on Machine Learning for Social Robotics, ICRA*, 2015.

[20] M.R.G. Meireles, P.E.M. Almeida, and M.G. Simoes. A comprehensive review for industrial applicability of artificial neural networks. *Industrial Electronics, IEEE Transactions on*, 50(3):585–601, June 2003.

[21] J.A.F.S. Pingenot. Decision tree learning, January 8 2015. US Patent App. 14/314,517.

[22] Mu Qiao and Jia Li. Two-way gaussian mixture models for high dimensional classification. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 3(4):259–271, 2010.

[23] Alina Roitberg, Alexander Perzylo, Nikhil Somani, Manuel Giuliani, Markus Rickert, and Alois Knoll. Human activity recognition in the context of industrial human-robot interaction. In *Asia-Pacific Signal and Information Processing Association, 2014 Annual Summit and Conference (APSIPA)*, pages 1–10, Siem Reap, Cambodia, December 2014. IEEE.

[24] Stephanie Rosenthal, Manuela Veloso, and AnindK. Dey. Acquiring accurate human responses to robots' questions. *International Journal of Social Robotics*, 4(2):117–129, 2012.

[25] Stephanie Rosenthal, Manuela Veloso, and AnindK. Dey. Is someone in this office available to help me? *Journal of Intelligent & Robotic Systems*, 66(1-2):205–221, 2012.

[26] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009.

[27] T.P. Spexard, M. Hanheide, and G. Sagerer. Human-oriented interaction with an anthropomorphic robot. *Robotics, IEEE Transactions on*, 23(5):852–862, Oct 2007.

[28] Lee Thomason. TinyXML-2. `http://www.grinninglizard.com/tinyxml2/index.html`.

[29] E.A. Topp and H.I. Christensen. Tracking for following and passing persons. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 2321–2327, Aug 2005.

[30] E.A. Topp, H. Huettenrauch, H.I. Christensen, and K. Severinson Eklundh. Bringing together human and robotic environment representations - a pilot study. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 4946–4952, Oct 2006.

[31] Elin Anna Topp. Understanding spatial concepts from user actions. In *Proceedings of the 6th international conference on Human-robot interaction*, pages 271–272, Lausanne, Switzerland, March 2011. ACM.

[32] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511–I–518 vol.1, 2001.

[33] Anna-Lisa Vollmer, Katrin Solveig Lohan, Kerstin Fischer, Yukie Nagai, Karola Pitsch, Jannik Fritsch, Katharina J Rohlfing, and Britta Wredek. People modify their tutoring behavior in robot-directed interaction for action learning. In *Development and Learning, 2009. ICDL 2009. IEEE 8th International Conference on*, pages 1–6. IEEE, 2009.

[34] Guorong Xuan, Wei Zhang, and Peiqi Chai. EM algorithms of gaussian mixture model and hidden markov model. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, volume 1, pages 145–148 vol.1, 2001.

Appendices

Appendix A
How to use the prototype

The software implemented in this project could be used as a base for future studies. Therefore, in this appendix we will explain how to install and run the prototype. This prototype software is licensed under MIT (`http://opensource.org/licenses/MIT`), so feel free to use, modify and improve.

A.1 Installation

Dependencies

This prototype has been implemented in C++ and ROS [7]. Besides, it requires SMILE and SMILearn libraries [2] for the Bayesian network. Hence, make sure ROS and the decision-theoretic model tools are installed in your computer.

Downloading packages

All the packages are part of the same metapackage that can be downloaded from github. Go to your ROS workspace and type the following to download the metapackage:

```
$ git clone https://github.com/FelipMarti/managing_interaction_patterns.git
```

Compiling packages

All the packages have to be compiled, to compile all of them type in your catkin workspace:

```
$ catkin_make
```

or specify the metapackage name:

```
$ catkin_make --only-pkg-with-deps managing_interaction_patterns
```

A.2 Running Interaction Patterns Manager prototype

Running packages to train a Bayesian network

Open one terminal and execute a roscore:

```
$ roscore
```

Open another terminal and launch the interaction_learner node. This node requires a Bayesian network structure as an input in the interaction_recognition/bayesian_network folder. The structure of the Bayesian network can be generated with GeNIe.

```
$ roslaunch interaction_learner interaction_learner_node.launch
BN:=test.xdsl
```

Open another terminal and launch the interaction_monitor node.

```
$ roslaunch interaction_monitor interaction_monitor.launch
```

Open another terminal and launch the elan_translator package. This roslaunch launches also the data_parser node, and requires as an input a set of annotations ELAN files in the data_parser/data folder.

```
$ roslaunch elan_translator elan_translator_node.launch
DATA:="subject1.eaf subject3.eaf subject5.eaf subject7.eaf"
```

Once the interaction_monitor finishes publishing all the annotations a new Bayesian network is generated "naivebayes.xdsl" in the interaction_recognition/bayesian_network folder.

Running packages to perform inference in a trained Bayesian network

Open one terminal and execute a roscore:

```
$ roscore
```

Open another terminal and launch the interaction_recognition node. This node requires a Bayesian network trained as an input in the interaction_recognition/bayesian_network folder.

```
$ roslaunch interaction_recognition interaction_recognition_node.launch
BN:=naivebayes.xdsl
```

Open another terminal and launch the interaction_monitor node.

```
$ roslaunch interaction_monitor interaction_monitor.launch
```

Open another terminal and launch the elan_translator package. This roslaunch launches also the data_parser node, and requires as an input a set of annotations ELAN files in the data_parser/data folder.

```
$ roslaunch elan_translator elan_translator_node.launch
```

```
DATA:="subject2.eaf subject4.eaf subject6.eaf subject8.eaf"
```

While the interaction_monitor publishes the annotations, the inference performed to the Bayesian network is printed. Once it stops publishing annotations, a statistical overview is printed with all of the matches, mismatches and unclear situations.

A.3 Running User movements from the Tracker prototype

Open one terminal and execute a roscore:

```
$ roscore
```

Open another terminal and launch the tracker_patterns node. This node requires as an input an odometry file in the tracker_patterns/data/odom folder and a trajectory file in the tracker_patterns/data/traj folder.

```
$ roslaunch tracker_patterns tracker_patterns_node.launch
"DATA:=P10odom.m P10traj_2.m"
```

While the tracker_patterns is running, it publishes the user movement in two different topics.

Can we understand people with the help of probabilistic methods?

Felip Martí Carrillo

Abstract— **By using verbal and non-verbal communication that the user unconsciously applies when interacting with a robot, we want to determine automatically what the user is trying to present.**

I. MOTIVATION

When people communicate and interact with others, a lot of non-verbal communication is produced that can help in the interpretation and understanding of the message. This non-verbal communication, among others, includes body language such as gestures, and the distance between interlocutors.

In the robotics world, one of the most challenging problems is to make robots able to understand humans. Therefore, if robots were able to interpret not only the verbal communication, but also the non-verbal one, robots would be more intelligent and would be able to disambiguate unclear situations that could happen during the interaction.

A previous study about Human-Robot Interaction, where people were presenting different rooms and items in an office environment, suggested that people generate some patterns while interacting with others that are different depending on the item that is being introduced or presented to the interlocutor. Furthermore, those patterns are quite common independently of the person that is communicating or interacting.

This led us to use probabilistic methods in order to automatically understand people when they are interacting with a robot, using all the patterns that are generated during the communication.

II. APPROACH

We want to analyse and test if we can understand people, with the help of probabilistic models, using all the patterns that people generate during the interaction. To do so, we have used all the data recorded in a previous study. These data are composed by manual annotations of the users' behaviour such as gestures, movements, instructions given to the robot, etc; and the robot sensor recordings.

Our approach stores from different sources of data all the user's behavioural features that occur around every item presentation to the robot, and it tries to recognise the item category using a probabilistic model (Bayesian Network).

We have divided all the items in three categories:

- **Workspace:** Specific positions/areas that can represent the position of large objects that are considered static. For example a coffee machine, a refrigerator or a printer.
- **Region:** Any portion of space that is large enough to allow for different workspaces in it, or at least large enough to navigate in it. Typically this would be rooms, corridors or parts of those.
- **Object:** Small items that can be handled by a human.

We also extract behavioural features from the movement detected by the sensor incorporated to the robot, this sensor is a laser scanner.

III. RESULTS

At around 70% of the cases, we succeeded understanding (or we were very close) what people presented, using as an input all the interaction patterns annotated in a manual analysis effort.

We got about 13% - 24% of mismatching cases that should be studied in more detail when trying to recognise between object/workspace or workspace/region. However, at least 50% of the mismatching cases were produced when classifying items that we previously did not have a clear classification for them.

What is a chair? Is it really an object or a workspace? We have realised during this study that there are a lot of objects/workspaces that we cannot really decide which is the most suitable category.

The extraction of behavioural features from the sensor on the robot worked as expected. However, the results obtained were different than the ones obtained in a manual effort.

Annotations from files are very subjective, even a small movement can be considered as a behavioural feature. That is not the case of our implementation, our algorithm is very rigid and only has two thresholds to detect movement features.

The results obtained are not perfect, but are very promising, and open a door to future research. Our approach has been designed considering future sources of data in order to easily extend this work.