Joint Perceptual Learning and Natural Language Acquisition for Autonomous Robots

Muhannad A R I Al-Omari

Submitted in accordance with the requirements

for the degree of Doctor of Philosophy

The University of Leeds

School of Computing

August 2017

Dedication

To my mom and dad who made me who I am,

and Dina who liked me that way.

Declaration

The candidate confirms that the work submitted is his/her own, except where work which has formed part of a jointly authored publication has been included. The contribution of the candidate and the other authors to this work has been explicitly indicated below. The candidate confirms that appropriate credit has been given within the thesis where reference has been made to the work of others.

Some parts of the work presented in this thesis have been published in the following articles. The publications are primarily the work of the candidate. Except where otherwise noted.

Alomari, M., Duckworth, P.[1]**, Bore, N., Hawasly, M., Hogg, D. C. and Cohn, A. G.** Grounding of Human Environments and Activities for Autonomous Robots. In 26th International Joint Conference on Artificial Intelligence, (IJCAI) , 2017.

Alomari, M., Duckworth, P., Hogg, D. C. and Cohn, A. G. Natural Language Acquisition and Grounding for Embodied Robotic Systems. In Association for the Advancement of Artificial Intelligence (AAAI), 2017.

Alomari, M., Duckworth, P., Hogg, D. C. and Cohn, A. G. Learning of Object Properties, Spatial Relations, and Actions for Embodied Agents from Language and Vision. In Spring Symposium of the Association for the Advancement of Artificial Intelligence (AAAI), 2017.

Alomari, M., Duckworth, P., Hawasly, M., Hogg, D. C. and Cohn, A. G. Natural Language Grounding and Grammar Induction for Robotic Manipulation Commands. In RoboNLP workshop in the Annual Meeting of the Association for Computational (ACL), 2017.

[1] Joint first author

Hawes, N., Burbridge, C., Jovan, F., Kunze, L., Lacerda, B., Mudrov a, L., Young, J., Wyatt, J. L., Hebesberger, D., K ortner, T., Bore, N., Ambrus, R., Folkesson, J., Jensfelt, P., Beyer, L., Hermans, A., Leibe, B., Aldoma, A., Faulhammer, T., Vincze, M. Z. M., Al-Omari, M., Chinellato, E., Duckworth, P., Gatsoulis, Y., Hogg, D. C., Cohn, A. G., Dondrup, C., Fentanes, J. P., Krajn k, T., Santos, J. M., Duckett, T., and Hanheide, M. The STRANDS project: Long-term autonomy in everyday environments. To appear In IEEE Robotics and Automation Magazine , 2017. As a member of the STRANDS European project, my contribution was limited to the development of activity recognition and human detection systems and their deployment on robots.

Alomari M., Duckworth, P., Gatsoulis, Y., Hogg, D. C., and Cohn, A. G. Unsupervised Natural Language Acquisition and Grounding to Visual Representations for Robotic Systems. In Proceedings of Cognitum Workshop, at IJCAI , 2016.

Gatsoulis, Y., Alomari M., Burbridge, C., Dondrup, C., Duckworth, P., Lightbody, P., Hanheide, M., Hawes, N., Hogg, D. C., and Cohn, A. G. QSRLib: a software library for online acquisition of qualitative spatial relations from video. In Proceedings of 29th Qualitative Reasoning Workshop, at IJCAI , 2016. My contribution in QSRlib was in designing and implementing the visualisations of QSRs into the library.

Alomari M., Chinellato, E., Gatsoulis, Y., Hogg, D. C., and Cohn, A. G. Unsupervised Grounding of Textual Descriptions of Object Features and Actions in Video. In Proceedings of the 15th International Conference on Principles of Knowledge Representation and Reasoning, (KR), 2016.

Acknowledgements

I gratefully acknowledge the financial support provided by the STRANDS project (Spatio-Temporal Representations and Activities for Cognitive Control in Long-Term Scenarios) funded by the European Community's Seventh Framework Programme, Cognitive Systems and Robotics, project reference 600623. I would like to thank colleagues in the School of Computing at the University of Leeds and in particular Paul Duckworth, Jawad Tayyub, Aryana Tavanai, Majd Hawasly, Yiannis Gatsoulis, Eris Chinellato, Krishna Sandeep Dubba, Muralikrishna Sridhar, and the STRANDS project consortium (http://strands-project.eu) for their valuable input and direction. Finally, I would especially like to thank my supervisors Professors Anthony Cohn and David Hogg for their guidance and support throughout my PhD.

Abstract

Understanding how children learn the components of their mother tongue and the meanings of each word has long fascinated linguists and cognitive scientists. Equally, robots face a similar challenge in understanding language and perception to allow for a natural and effortless human-robot interaction. Acquiring such knowledge is a challenging task, unless this knowledge is pre-programmed, which is no easy task either, nor does it solve the problem of language difference between individuals or learning the meaning of new words. In this thesis, the problem of bootstrapping knowledge in language and vision for autonomous robots is addressed through novel techniques in grammar induction and word grounding to the perceptual world. The learning is achieved in a cognitively plausible loosely-supervised manner from raw linguistic and visual data. The visual data is collected using different robotic platforms deployed in real-world and simulated environments and equipped with different sensing modalities, while the linguistic data is collected using online crowdsourcing tools and volunteers. The presented framework does not rely on any particular robot or any specific sensors; rather it is flexible to what the modalities of the robot can support.

The **learning framework** is divided into three processes. First, the perceptual raw data is clustered into a number of Gaussian components to learn the 'visual concepts'. Second, frequent co-occurrence of words and visual concepts are used to learn the language grounding, and finally, the learned language grounding and visual concepts are used to induce probabilistic grammar rules to model the language structure.

In this thesis, the **visual concepts** refer to: (i) people's faces and the appearance of their garments; (ii) objects and their perceptual properties; (iii) pairwise spatial relations; (iv) the robot actions; and (v) human activities. The visual concepts are learned by first processing the raw visual data to find people and objects in the scene using state-of-the-art techniques in human pose estimation, object segmentation and tracking, and activity analysis. Once found, the concepts are learned incrementally using a combination of techniques: Incremental Gaussian Mixture Models and a Bayesian Information Criterion to learn simple visual concepts such as

object colours and shapes; spatio-temporal graphs and topic models to learn more complex visual concepts, such as human activities and robot actions.

Language grounding is enabled by seeking frequent co-occurrence between words and learned visual concepts. Finding the correct language grounding is formulated as an integer programming problem to find the best many-to-many matches between words and concepts.

Grammar induction refers to the process of learning a formal grammar (usually as a collection of re-write rules or productions) from a set of observations. In this thesis, Probabilistic Context Free Grammar rules are generated to model the language by mapping natural language sentences to learned visual concepts, as opposed to traditional supervised grammar induction techniques where the learning is only made possible by using manually annotated training examples on large datasets.

The learning framework attains its **cognitive plausibility** from a number of sources. First, the learning is achieved by providing the robot with pairs of raw linguistic and visual inputs in a "show-and-tell" procedure akin to how human children learn about their environment. Second, no prior knowledge is assumed about the meaning of words or the structure of the language, except that there are different classes of words (corresponding to observable actions, spatial relations, and objects and their observable properties). Third, the knowledge in both language and vision is obtained in an incremental manner where the gained knowledge can evolve to adapt to new observations without the need to revisit previously seen ones (previous observations). Fourth, the robot learns about the visual world first, then it learns about how it maps to language, which aligns with the findings of cognitive studies on language acquisition in human infants that suggest children come to develop considerable cognitive understanding about their environment in the pre-linguistic period of their lives. It should be noted that this work does not claim to be modelling how humans learn about objects in their environments, but rather it is inspired by it.

For **validation**, four different datasets are used which contain temporally aligned video clips of people or robots performing activities, and sentences describing these video clips. The video clips are collected using four robotic platforms, three robot arms in simple block-world scenarios and a mobile robot deployed in a challenging real-world office environment observing different people performing complex activities. The linguistic descriptions for these datasets are obtained

using Amazon Mechanical Turk and volunteers. The analysis performed on these datasets suggest that the learning framework is suitable to learn from complex real-world scenarios. The experimental results show that the learning framework enables (i) acquiring correct visual concepts from visual data; (ii) learning the word grounding for each of the extracted visual concepts; (iii) inducing correct grammar rules to model the language structure; (iv) using the gained knowledge to understand previously unseen linguistic commands; and (v) using the gained knowledge to generate well-formed natural language descriptions of novel scenes.

Acronyms and Symbols

API	Application Programming Interface
BIC	Bayesian Information Criterion
CFG	Context Free Grammar
CNN	Convolutional al Neural Network
CPM	Convolutional Pose Machine
DAG	Directed Acyclic Graph
DOF	Degrees Of Freedom
DR	Distinct-Regions
EM	Expectation Maximisation
$FPFH$	Fast Point Feature Histogram
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
HSL	Hue-Saturation-Lightness
$IGMM$	Incremental Gaussian Mixture Model
LDA	Latent Dirichlet Allocation
$LUCIE$	Leeds University Cognitive Intelligent Entity
$LUCAS$	Leeds University Cognitive Artificial System
NLP	Natural Language Processing
$NLTK$	Natural Language Tool-Kit
$PCFG$	Probabilistic Context Free Grammar
PO	Partially-Overlapping
POS	Part-of-Speech
PP	Proper-Part
SVM	Support Vector Machine
QDC	Qualitative Distance Calculus
QSR	Qualitative Spatial Representation
QTC	Qualitative Trajectory Calculus
RCL	Robot Control Language
$RCC5$	Region Connection Calculus 5
RGB	Red-Green-Blue
RGB-D	Red-Green-Blue-Depth
ROS	Robotics Operating System
$SLAM$	Simultaneous Localisation And Mapping
$TPCC$	Ternary Point Configuration Calculus

$\mathcal{A}$	Target function resultant from solving the integer program
$\mathcal{B}$	Non-terminal symbol
b	Number of all unique n-grams
c	*id* of a connection node in DAG
$\mathcal{C}$	Non-terminal symbol
d_M	Mahalanobis distance
G	A Gaussian component
G_R	*id* of robot gripper in a spatio-temporal DAG
$H(.)$	Entropy of a set
j	Human body part or joint
J	Human pose estimate vector comprising of 15 body parts
$\mathbf{K}$	Concepts correlation matrix
$\mathbf{K}(i,j)$	The element at the i^{th} row and j^{th} column in matrix $\mathbf{K}$
m	Number of objects in a video clip
M	Longest sequence of n words to form an n-gram
n_d	Number of dimensions in a feature space
n_f	Number of most prominent Eigen faces
n_h	Habituation constant
N	Set of non-terminal symbols
$\mathcal{N}$	List of all observed n-grams
p	Pixel or point in a point cloud
P	Set of probabilities on production rules
R	Set of production rules
S	Start symbol in a grammar
T	Set of terminal symbols
u	Number of all unique visual concepts
v_i	A visual concept
V_m	V-measure used in evaluating clustering techniques
$\mathcal{V}$	List of all observed visual concepts
X	Observation vector in a visual feature space
y_∞	Limiting value in the exponential function
$\mathcal{Z}$	A terminal symbol
α	Constant in the exponential function
δ	A single n-gram
μ	Mean of a Gaussian component
ϵ	Threshold to keep sparsity of groundings
τ	Decaying rate constant
$\lambda(.)$	Counting function
Σ	Covariance matrix of a Gaussian component
Φ	Language grounding function
Ω	Vision tree
Ψ	RCL tree
Π	Language grammar

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

"Instead of trying to produce a programme to simulate the adult mind, why not rather try to produce one which simulates the child's? If this were then subjected to an appropriate course of education one would obtain the adult brain."

—*Alan Turing, 1950*

1.1 Motivation

To date, the human brain is the only known system capable of fully learning and understanding natural language. Understanding the process though which we learn our mother tongue has received a great deal of attention since the 1950s in various fields from psychology and cognitive sciences to computer sciences. In computer science, researchers were particularly interested in understanding this learning process in order to transfer it to a machine. Alan Turing (1950) was one of the earliest researchers to articulate this idea by suggesting to provide the machine with the ability to learn like a child, rather than providing it with the knowledge itself. This way, a machine could theoretically learn in the same manner as humans do resulting in the same knowledge if provided with the same course of education. Although this idea remains as an ambition for the future, in this work we explore parts of it. We aim to provide a machine with the ability to bootstrap its knowledge in natural language and perception. By doing so, we provide robotic agents with the ability to learn from their own experiences about people, objects and activities in their environments. It must be noted that even though our learning system is inspired by how children learn about their language, we do not claim that this work models the same learning procedure.

1.2 The Language Acquisition Problem

The language acquisition problem, be it for humans or robots, has received a great deal of attention over the years. The quest to find solutions to this problem has raised many questions, some concerning the nature of how we learn our language, others concerning transferring this learning ability to a machine. In this work, we attempt to tackle some of these questions by addressing the problem of joint perceptual learning and natural language acquisition for autonomous robots. The bulk of this thesis focuses on the loosely-supervised learning of a language's syntax and semantics from a corpus of videos and descriptions featuring robots and humans performing various tasks.

1.2.1 Language acquisition in robotics

For robots to integrate in human environments, it is essential that they be equipped with the ability to continuously learn about their environments, the people that inhabit these environments, the activities that take place in it, and how to perform useful tasks. From an autonomous robot point of view, this requires incremental learning methods that can operate on the outputs of various sensing modalities, such as RGB and depth cameras, voice recognition, laser rangefinder measurements, etc. The outcome of this learning process is a collection of concepts, such as objects, people, relations and activities that occur in the robot's environment, as well as their mapping to natural language such that the robot can understand given commands and be able to interact with humans.

Researchers have tackled the language acquisition problem for robotic systems using different approaches, such as *individual* and *social learning*. In *individual learning*, the robot is provided with data to learn about natural language without any further assistance from a teacher, and is expected to learn from such data as presented by Siskind (1996), Roy *et al.* (1999) and Needham *et al.* (2005). In *social learning*, the teacher plays an important role in the learning process, by providing feedback to guide the learner in acquiring different language components as presented by Steels & Kaplan (2002) and Spranger (2015). In this research, we follow the *individual approach* as it enables learning from large datasets without the need for constant supervision, which is more suitable for autonomous life-long learning for robots.

1.2.2 Language acquisition in cognitive sciences

Over the last forty years, more evidence has emerged supporting the idea that language acquisition process relies significantly on our ability to learn cognitive concepts in the prelinguistic period of our lives. However, this was not always the case. In the 1950s and 1960s, the general assumption was that children learn semantic categories of their environment using language. As suggested by Whorf (1956), language shapes our understanding of the world by noticing which properties of referents remain constant across successive uses of a word by a fluent speaker. In the early 1970s, the idea of how we learn language began to change as more evidence showed that Whorf's hypothesis is incorrect. A number of studies on language acquisition in human infants such as ones presented by Piaget (1954), Berlin and Kay (1969), Rosch and Mervis (1977) and Bowerman (1996) have suggested that children come to develop considerable cognitive understanding about their environment in the prelinguistic period of their lives. In other words, these studies suggest that humans come to learn about object properties, spatial relations and other cognitive concepts before they learn the words used to describe them in natural language. The work presented in this thesis aligns with those findings, whereby the robots develop a cognitive understanding of their environments before attempting to ground the words used to describe them in natural language.

1.3 Learning Framework

This work aims to answer the following two questions, (i) can a robot bootstrap its knowledge in language and vision? and (ii) can a robot ground language to concepts in vision? To answer these questions in a cognitively plausible setting, we take inspiration from human learning, which is incremental and is typically loosely supervised. Furthermore, our system is tasked with learning incrementally from human description of the world, while the outcome of the learning process is representable in a human understandable form. We present a novel individual learning approach capable of acquiring symbolic knowledge in both language and vision concurrently, and use this knowledge to parse previously unseen natural language commands. The learning is accomplished using a show-and-tell procedure; this is inspired by how children acquire knowledge of their everyday physical world by interacting with their parents. The learning data comes from two sources, (a) volunteers controlling a robot to perform a variety of table top tasks, (b) people performing everyday activities in an office environment. Both were subsequently annotated with appropriate linguistic descriptions as shown in Figure 1.1. The recorded videos and descriptions are used as input data to our system to learn three key components, (*i*) the visual representation of the world; (*ii*) the groundings of words and phrases to the learned visual representations; and (*iii*) the grammar rules of the language. To the best of our knowledge, this is the first system that learns these three components concurrently.

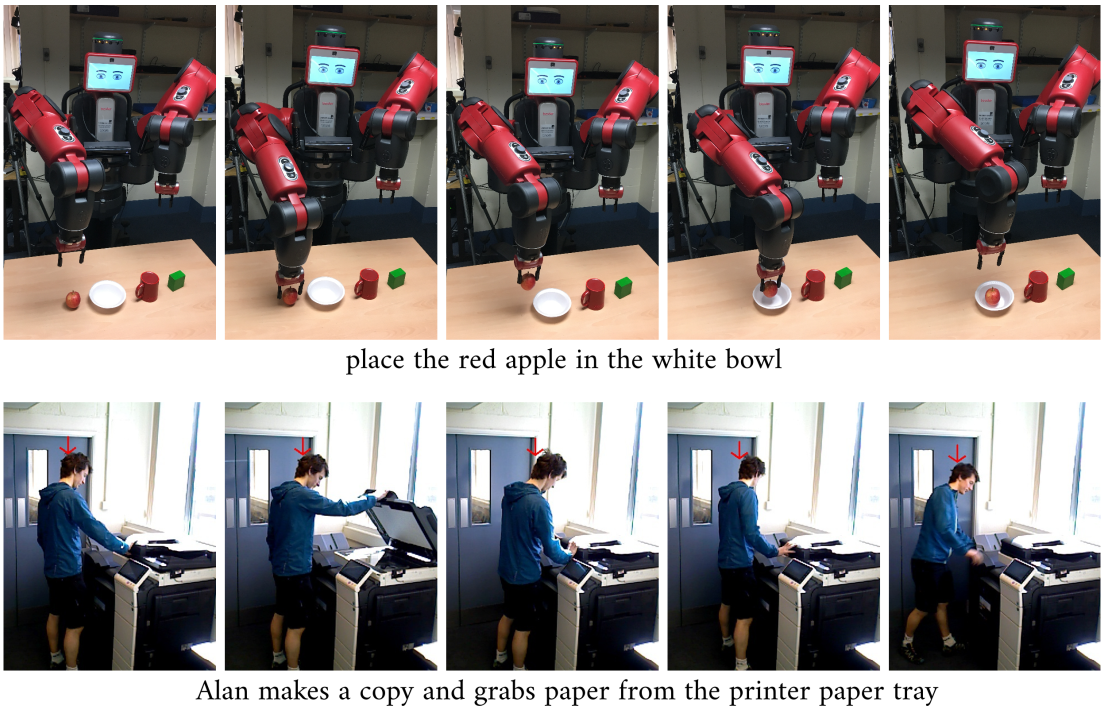

place the red apple in the white bowl
Alan makes a copy and grabs paper from the printer paper tray

Figure 1.1: Examples of input video clips annotated with the natural language commands.

We presuppose that the robots can visually analyse the environment in order to extract a multitude of features and incrementally recover useful classes of features, which are referred to as *visual concepts*: abstractions of the feature spaces generated by the robot modalities which carry a human-level meaning, for example the colour red, or the face of a person. Once we learn the visual concepts, the natural language descriptions are analysed to ground words and phrases to their most relevant visual concepts, followed by learning simple syntactic rules (i.e. a grammar) that govern the sentence structure. Thus, the framework supports recognition of objects, individuals, spatial relations, robot actions and human activities. To do this we integrate state-of-the-art object segmentation, pose estimation and activity analysis into a flexible, incremental framework. We aim to learn and distinguish instances of human-level visual concepts in simulated and real-world complex scenarios in a loosely supervised manner. The learning framework shown in Figure 1.2 represents our entire system for bootstrapping knowledge in natural language and perception for robotic systems.

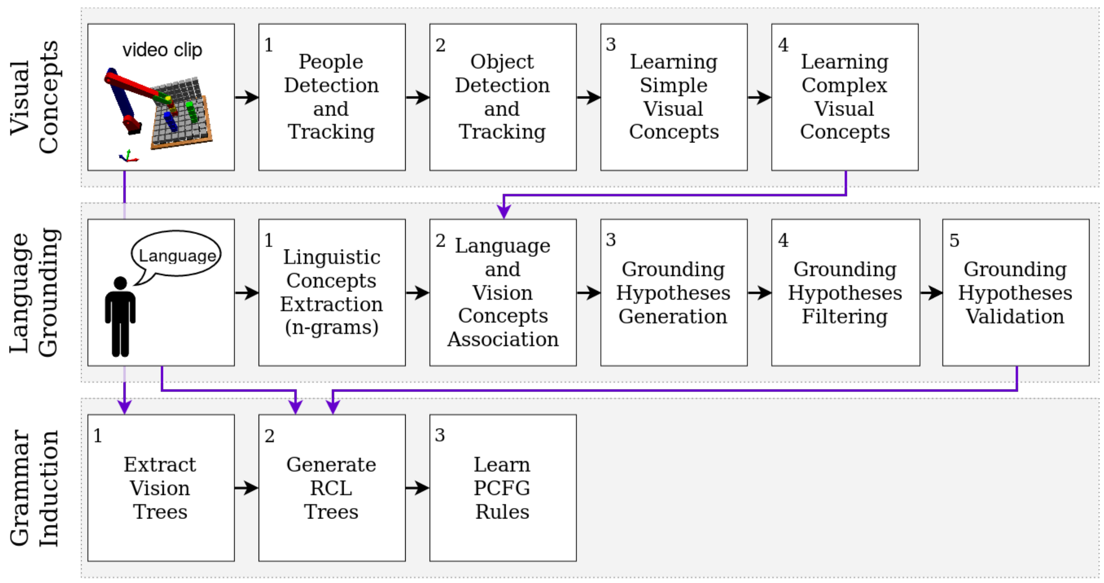

Visual Concepts
video clip
1 People Detection and Tracking
2 Object Detection and Tracking
3 Learning Simple Visual Concepts
4 Learning Complex Visual Concepts
Language Grounding
Language
1 Linguistic Concepts Extraction (n-grams)
2 Language and Vision Concepts Association
3 Grounding Hypotheses Generation
4 Grounding Hypotheses Filtering
5 Grounding Hypotheses Validation
Grammar Induction
1 Extract Vision Trees
2 Generate RCL Trees
3 Learn PCFG Rules

Figure 1.2: Language and vision learning framework. Consisting of three main components: learning of visual concepts, natural language grounding and grammar induction.

The framework consists of three main components: 1-Learning Visual Concepts, 2-Natural Language Grounding, and 3-Grammar Induction. The framework is applied on every video-sentence input, and the cumulative knowledge in language and vision is updated incrementally with each processed input. In visual concepts learning, we first perform low-level analysis on videos to detect and track objects and people in the scene. Then, we incrementally learn the visual representation of the world in a number of predefined feature spaces such as colours, people, object shapes, robot actions, etc. by clustering values observed from each video clip. These features are divided into two categories, *simple* and *complex* features based on their measurement and encoding complexities. Further details on learning visual concepts are provided in Chapter 3. In natural language grounding, we map words and phrases to their corresponding visual concepts. For example, we learn that the word 'red' is used to describe the colour red. The language grounding approach is discussed in detail in Chapter 4. In grammar induction, we learn simple syntactic rules that enables the robot to understand new linguistic commands. The learning of grammar rules is achieved by mapping the sentence structure to the structure of the input video, more details on our grammar induction approach are provided in Chapter 5.

1.4 Assumptions and Limitations

This work makes several assumptions, some of which are purposeful and serve to usefully delimit the scope of the investigation while others are more problematic and are left as open research questions for future work in language acquisition in robotics.

1.4.1 Loosely-supervised learning

We use the term *loosely-supervised* to describe the learning process that requires the videos and sentences to be temporally aligned beforehand, making it more suitable for teaching robots about basic concepts such as colours or shapes. We consider the learning in our system to be loosely-supervised rather than unsupervised. A fully unsupervised system would be able to learn from longer non-segmented videos and documents (i.e. be able to temporally segment and align long videos and documents), or even learn from a constant stream of audio-video data, which remains an ambition for the future.

1.4.2 Innate versus learned knowledge

In a dynamic environment, animals must constantly gain new information and skills to survive. However, in a stable environment, the same individuals need to gather the information needed once, and then rely on it to survive. Therefore, different environments better suit either the need for innate or learning knowledge. Essentially the cost of obtaining certain knowledge versus the benefit of already having it, determines whether an animal evolved to learn or to innately know the information, as presented by Mery and Kawecki (2004). The same line of thought can be applied to machines: if a machine is expected to repeatedly perform a single task, e.g. car assembly robots, then it is more beneficial to provide the machine with knowledge needed to perform this specific task. However, if it is expected to perform numerous tasks, e.g. home and office service robots, then gaining knowledge becomes more important for the utility and survival of the machine. We provide our robots with enough innate knowledge to enable them to learn useful concepts in their environments. For example, we assume the robots are capable of detecting and tracking both objects and people in each video using a number of state-of-the-art techniques. We also assume that each robot has a predefined set of visual feature spaces to learn from. Moreover, we assume that each robot has a set of predefined language classes that enables the learning of language grammar. Each of these assumptions limits the learning ability of our robots by adding a certain constraint. For example, the nature of objects we can detect and track are limited by the algorithm we use. However, we argue that having these assumptions allow our robots to focus on learning interesting concepts in both natural language and vision.

1.5 Thesis Structure

In this chapter, we briefly motivated the language acquisition problem for robotic systems, defined our learning framework and its three main components, and discussed the main assumptions and limitation of our work. The remainder of this thesis is presented as follows.

In Chapter 2, we present the literature review of the language acquisition problem, starting from one of the earliest language and vision systems in computing (SHRDLU 1972), up until state-of-the-art techniques, and discuss our system's contributions.

In Chapter 3, we present the visual concepts learning framework. We start by introducing the robotic systems used in this thesis, describe the low-level processes used to detect and track objects and people in each video, and finally present in detail how we learn the visual concepts. We also list the techniques used and the assumptions made for each of the robotics systems.

In Chapter 4, we present the language grounding framework, and in particular the details of techniques used to enable the mapping of natural language words and phrases to visual concepts.

In Chapter 5, we present the grammar induction process from linguistic and vision data, and elaborate on the algorithm we developed to enable the robots to learn simple grammatical rules, which are used to enable understanding and execution of new linguistic commands.

In Chapter 6, we present the experimental setup used to evaluate each of the framework's three components, namely, learning visual concepts, language grounding, and grammar induction. Four experiments are performed with different evaluation measures to examine the performance and scalability of our system. We also compare our results with other supervised and unsupervised techniques to better demonstrate our system's abilities and limitations.

In Chapter 7, we present the key findings of our work, and the contributions we offer in the field of natural language acquisition in robotics. We also discuss the main limitations of this work and suggest a number of research directions for future work.

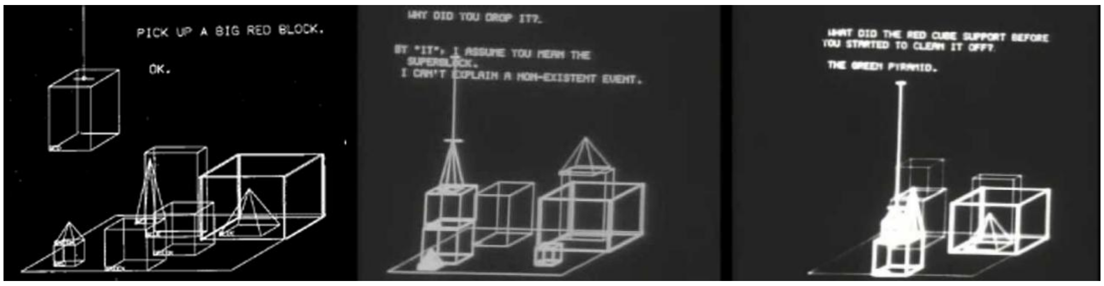
Chapter 2

Related Work

2.1 Background

Natural Language acquisition and understanding has been a long standing objective of AI and robotics research. One of the earliest computer systems capable of understanding natural language commands to perform simple tasks in a virtual world is 'SHRDLU' by Winograd (1972). It was pre-equipped with all the linguistic knowledge needed to understand and execute linguistic commands such as "*pick up the red block*", and answer questions such as "*what did the red cube support before you started to clean it off?*", as shown in Figure 2.1. Another system by Hogg (1977) was capable of generating linguistic descriptions, in real time, of simple visual situations involving one or two moving objects using picture differencing algorithm. The system was implemented on a DECsystem-10 and was capable of generating scene descriptions by processing images of people moving in the lab such as "*An object has appeared at left of scene, call it object A. Object A has begun moving. Object A looks like a person, call it Fred*".

Our system can incrementally acquire parts of the knowledge needed to perform similar tasks from real-world data. We focus on joint learning of perceptual and language components for autonomous robots. This chapter is divided into two sections: (i) Natural Language Acquisition in Robotics, and (ii) Perception in Robotics. In each section, we discuss relevant work done in the field and compare it with our system where applicable.


PICK UP A BIG RED BLOCK.

OK.

WHY DID YOU DROP IT?.

BY "IT", I ASSUME YOU MEAN THE SUPERBLOCK.
I CAN'T EXPLAIN A NON-EXISTENT EVENT.

WHAT DID THE RED CUBE SUPPORT BEFORE YOU STARTED TO CLEAN IT OFF?

THE GREEN PYRAMID.

Figure 2.1: SHRDLU system by Winograd (1972). It was capable of understanding a variety of commands that included object descriptions and table-top actions such as *pick up, move, etc.*

2.2 Natural Language Acquisition in Robotics

Language acquisition is the process by which humans acquire the knowledge needed to perceive and comprehend natural language, as well as to produce meaningful words and sentences to communicate with others. The field of language acquisition contains within it a large number of research areas. We focus on learning only two aspects of language in this work. The grounding of language to vision (learning a semantic representation of language), and the grammar rules of language (learning a syntactic representation of language). These two components are essential for understanding simple natural language commands such as "*pick up the red block*", and therefore, are a good starting point to bootstrap our robot's knowledge in natural language. In the following sections, we discuss the recent work done in language grounding and grammar induction, and their application in understanding natural language commands.

2.2.1 Language grounding

In the field of **psychology**, *referential uncertainty* is often thought to be the most important aspect of word learning. As described by Quine (1960), referential uncertainty is defined as the problem of how words get their meanings, which is a general problem everyone faces when trying to learn a new language. The space of possible meanings for a new word is infinite. A word can be used to refer to anything from a physical object (e.g. book), a feeling (e.g. love), a mathematical process (e.g. integration), etc. The language grounding problem can be thought of as a subset of the referential uncertainty problem, where the space of possible meanings are limited to concrete observable ones.

We limit the space of possible meanings of a word to ones that can be measured using the robot's limited sensing modalities, such as objects' shapes and colours, spatial relations, peoples' appearances, etc. The aim of language grounding is to learn the words used to refer to each of these meanings. For example, learning that the word '*red*' refers to a particular connected subset of the HSL colour space.

In **computer science**, Siskind (1996) was one of the earliest researchers to try and understand how children ground their language to vision in a computational setting. His research focused on understanding how children learn their native language, and how their language is mapped to their visual representation of the world; however, he separated the learning of language from learning the mapping of words to visual concepts. Following his research, one of the earliest works to try and learn language grounding for robotics applications was a system by Roy *et al.* (1999), their system was capable of learning audio-visual associations (basically objects' names) using mutual information criteria from recorded images and audio data. Several robotic applications were developed subsequently, such as Steels (2001) where language games were used to teach autonomous robots the meaning of words. Two pan-tilt cameras looking at a white board containing coloured geometric figures were used as robots to learn the words used to refer to colours and shapes through a guessing game. Further, Needham *et al.* (2005) used language grounding as part of a system to teach artificial agents to play table-top card games. The system observed two people playing the game, and recorded audio-video data of how the game is played, and the names of the different cards, which were used to teach the system how to play the game and interact with other players.

Researchers also used **social learning** to teach robots the language grounding in vision. For example, the work by Steels & Kaplan (2002) "*AIBO's First Words*" designed a language grounding framework that enabled a robot dog, called AIBO, to learn the meaning of words through social interaction. The learning commenced by presenting AIBO with a small object while the teacher utters the object's name; this provided the robot with the audio-visual data needed to learn the language grounding. Later, the teacher presented the same object to AIBO and asked "*What is it?*". If AIBO answers correctly, the teacher says "*Good.*", otherwise, the teacher corrects AIBO by repeating the object's name again. Following their work, Bleys *et al.* (2009) developed a colour naming game for robots. The game was played with two humanoid robots, where one of them teaches the other the names of different colours by pointing and uttering the names in a static environment. Guadarrama *et al.* (2013) focused on teaching a PR2 robot (2010) spatial relations through interacting with a human teacher. The gained knowledge allowed the robot to execute linguistic commands with spatial prepositions, such as "*Move the cup close to the robot to the area in front of the plate and behind the tea box*". However, the knowledge about language grammar and actions was provided to the robot beforehand to enable the learning. Similarly, Lanbo She *et al.* (2014) implemented a system that can teach a robot arm simple manipulation actions by having a dialogue with expert-users. The system grounded words to sequence of predefined atomic actions using natural language. For example, to teach the arm how to grab an object the following dialogue was used: Human: "*Grab the blue block.*" Robot: "*What do you mean by grab?*" Human: "*Open your gripper.*" Robot: "*Ok.*" Human: "*Move to the blue block.*" Robot: "*Sure.*" Human: "*Close gripper.*" Robot: "*Alright then.*" Human: "*Now you achieved the grab action.*" Robot: "*Ok, got it.*". Their system was able to ground words to sequences of predefined primitive actions with the assumption that the robot knows the objects and how to ask and answer questions when speaking to a user. Further, the work of Spranger & Steels (2015) focused on teaching a robot about spatial relation's utterances in guided-learning interactions with a tutor robot. The tutor robot was equipped with a system for producing English spatial phrases, and is responsible for guiding the language grounding process by simplifying the challenges and complexities of utterances, providing feedback to the student robot, and gradually increasing the complexity of the language to be learned. While Spranger focused on robot teachers, Parde *et al.* (2015) focused on enabling non-expert users to teach robots about object names using a game called *I Spy*: a guessing game where the spy says "*I spy with my little eye (object name)*" and players have to guess the object the spy saw. In their work, the player was replaced by a robot trying to learn object names by playing the game. Their system filtered out unwanted words using a stop-words list in order to extract the key words in the input sentences, which are used in language grounding and learning object names. Hristov *et al.* (2017) presented a framework that exploits the pragmatics of human sensorimotor behaviour to derive cues that enable the grounding of symbols to object's colours and shapes. A 3D eye tracking sensor was used to track the non-expert user's gaze while performing a given task such as "*put the red cube on top of the yellow cube*". The 3D tracking sensor provided the robot with data needed to detect and track objects in the scene, as well as hints to which objects were being described in the given input sentence since the users were more likely to focus on them.

Researchers also used **web-available** descriptions and images to teach robots how to perform different tasks. For example, Beetz *et al.* (2011) implemented a system that used descriptions from the *wikihow* website[1] to teach a robot to make pancakes in their work "*Robotic Roommates Making Pancakes*". The descriptions included the procedure of how to make pancakes. Examples of these descriptions included instructions like "*1-Take the pancake mix from the refrigerator*", "*2-Add 400ml of milk (up to the marked line) shake the bottle head down for 1 minute. Let the pancake-mix sit for 2-3 minutes, shake again*", "*3-Pour the mix into the frying pan*", etc. To understand such instructions, the robot needs to link the steps to the appropriate predefined atomic actions in its library and ground the abstract ingredient and utensil descriptions with their corresponding objects in its environment. Similarly, Dubba *et al.* (2014) presented a system that teaches a robot to arrange a casual dinner table. A PR2 robot learned to arrange cutlery and plates on a table, acting as a waiter. The learning was achieved using web-available descriptions from the *wikihow* website[2] in the form of a sequence of instructions. For example, "*1-Set a placemat on the table*", "*2-Arrange your plate and napkin*", "*3-Place your silverware on the placemat*", etc. The descriptions were used to learn objects' arrangements on the table using language grounding to vision.

In the field of **robot navigation**, Lauria *et al.* (2002) introduced a system capable of teaching a vision based miniature mobile robots to navigate inside a miniature town. Their system used natural language commands for learning, such as "*take the second right after the post-office*". The system learned the meaning of each word by grounding it to predefined primitive procedures. Following their research, Huang *et al.* (2010) developed a supervised system capable of guiding a drone in a 3D environment. The system used natural language commands to learn from, such as "*Fly past room 124 and then face the windows. Go up. Go back down*". The system learned meanings of words by grounding them to previously labelled actions and objects in the environment. Tellex *et al.* (2011) developed a system that grounds

[1]http://www.wikihow.com/Make-Pancakes-Using-Mondamin-Pancake-Mix
[2]http://www.wikihow.com/Set-a-Dinner-Table

words and phrases to predefined objects (e.g. a truck or a door), places (e.g. a particular location in the world), paths (e.g. a trajectory through the environment), and events (e.g. a sequence of predefined atomic robot actions). Their system aimed to guide a forklift using natural language commands, such as "*Put the tire pallet on the truck*". Mutsaziek *et al.* (2013) implemented a system capable of parsing natural language commands to actions and control structures that can navigate a mobile robot in an office environment. The system learned the meaning of words by grounding them to predefined location nodes, relations and actions in a supervised probabilistic manner. For example, it grounded the word 'left' to a predefined relation of the direction left. This enabled the robot of executing commands like "*Go straight down the hallway past a bunch of rooms until you reach an intersection with a hallway on your left; turn left there*". Thomason *et al.* (2015) implemented an agent that expands its natural language understanding incrementally from conversations with users. The system learned new words by grounding them to predefined directory of people, objects and offices. Their system assumed that a robot can have a conversation with users to ask about the meaning of the unknown words. Hemachandra *et al.* (2015) presented a system that uses language grounding techniques to understand navigation commands. Their goal was to enable a user of guiding a wheelchair with voice commands, such as "*Go to the kitchen that is down the hallway*". To do so, the mapping between the given commands and the environment had to be learned. The learning was achieved by grounding words to parts of a graphical model used to represent the world map and actions. Barrett *et al.* (2017) presented a framework which supports grounding the semantics of natural language in the domain of robot navigation. Their system focused on learning meanings of nouns and prepositions from noisy data containing sentences describing paths driven by a robot, such as "*The robot went right of the table which is left of the chair, then went in front of the chair, then went behind the table which is right of the chair*". The gained knowledge was used to enable their robot of executing novel commands and drive in new paths. However, they assumed predefined object types and actions to enable the learning.

In the field of **robot manipulation**, researchers used artificial neural networks to ground language in robot actions. For example, Wermter & Elshaw (2003) presented a system based on self-organising maps approach that learns to control a robot using language instructions. The system takes as inputs the verb from the input sentence, and the sensory-motors data from the robot. A self-organising map is used to map these two inputs together to learn the meanings of verbs in robot actions. However, they limited the language grounding to verbs.

In the works described above, the language grounding problem was simplified by using one of the four following assumptions. The robot was assumed to have the knowledge of: 1-A stop word list to filter out unwanted words, such as function words: a word whose purpose is to contribute to the syntax rather than the meaning of a sentence, for example '*do*' in "*we do not live here*". 2-The syntactic or semantic grammar rules, used to parse input sentences and extract key words such as verbs, nouns, etc. rather than learning from raw textual descriptions. 3-A set of predefined atomic actions, spatial relations, or object classes that are used as the space of possible meanings of language, rather than learning from raw vision data. 4-A teacher that supervised the learning of language grounding and provides constant feedback to correct any mistakes. These assumptions simplify the learning of language grounding and enables the robot to focus on learning more complex concepts, such as making pancakes. However, in this work, we focus on natural language acquisition itself, and present a novel technique capable of acquiring semantic meanings of words and phrases from unlabelled linguistic and vision data. We improve on the above mentioned works in a number of ways by tackling the same learning problem using a more relaxed set of constraints. First, we assume less predefined knowledge is available to the robot initially. For example, we do not assume having the knowledge of a stop word list initially, or knowing the language grammar beforehand. Second, we learn the space of visual meanings by clustering features in videos, as opposed to assuming a set of predefined objects and colours, or having a list of predefined atomic actions that the robot knows of. Third, we learn from real-world noisy data, allowing multiple objects to be present in the scene during learning, and allowing for objects to be partially occluded.

2.2.2 Grammar induction

Having a working knowledge of language grammar is essential for understanding the meaning conveyed by sentences, and developing our ability to express and respond to this meaning. Grammar induction refers to the process of learning a formal grammar from a set of observations, usually as a collection of re-write rules or productions or alternatively as a finite state machine or automaton of some kind. Thus, constructing a model for the syntactic or semantic structure of natural language.

Learning Grammar rules has been a long standing objective in linguistic and cognitive studies. Researchers tackled the problem of how humans learn their grammar. The work of Hudson-Kam and Newport (2005) led to the idea that children learn their languages based on the frequency of the grammatical forms they hear. We adopt this approach and learn something of the wording meaning and grammatical structure of imperative sentences and the way in which this relates to objects, relations and actions depicted in video clips. However, it is worth noting that our system does not claim to be cognitively modelling human language learning, even though certain aspects of it may be cognitively plausible.

Researchers have tackled the grammar induction problem in a **supervised manner** to enable their robots to understand natural language commands. The robots were provided with input sentences along with their manually annotated grammar trees, be it syntactic or semantic trees. For example, MacMahon *et al.* (2006) presented a system that learns to follow verbal route instructions in a simulated world. The learning was achieved by inducing grammar rules that parse the natural language commands into a predefined structure that the robot can execute. The learning of these rules was performed in a supervised manner. The system used the learned rules to execute new commands such as "*With the wall on your left, walk forward*", or "*Walk to the further end of the hall*". Matuszek *et al.* (2013) used probabilistic *combinatory categorial grammars* (CCG) presented by Steedman (2000) to parse natural language navigation instructions such as "*exit the room and go left*" into a LISP-like tree representation that the robot can understand and execute such as "(*do-sequentially* (*take-unique-exit*) (*turn-left*))". The grammar rules learned from the training data enabled their mobile robot of executing new linguistic commands, provided that actions (e.g. *turn*), spatial relations (e.g. *left*) and locations on the map (e.g. *room*) were predefined initially and known to the robot. Similarly, Dukes (2014) learned how to parse natural language commands for manipulation tasks into a formal representation named Robot Control Language (RCL): a tree semantic representation for natural language commands. Each sentence is represented as an RCL tree, where leaf nodes align to words in the corresponding sentence, and non-leaves are labelled with a predefined set of categories that the robot can understand. Dukes used supervised grammar induction technique to enable the robot of understanding natural language commands in a simulated world such as "*place the red tetrahedron to on top of the blue and green tower*", provided that colours, actions and spatial relations were predefined initially. Further, Wang *et al.* (2016) learned how to parse natural language commands by learning semantic parsing technique in a simulated world. The system is trained using a web-interface[3] where users played a game with a robot. The users provided the robot with commands such as "*remove all red*", and provided feedback until the robot succeeded in performing the given command. This generated training data needed to learn to parse natural language commands into a plan which the robot can understand and execute. However, their system assumed that object colours, relations and actions were predefined initially to enable the learning of grammar rules.

Unsupervised learning techniques were also used to tackle the grammar induction problem from unlabelled sentences. For example, Chen and Mooney (2011) implemented a system that learns to transform natural-language navigation instructions into executable formal plans. The transformation from language to plans is achieved using a grammar parser that was trained without using direct supervision. However, the parser was provided with natural language instructions such as "*Place your back against the wall of the 'T' intersection. Turn left. Go forward*", and their human-annotated plans that the robot can understand and execute such as "Turn(), Verify(back:WALL), Turn(LEFT), Travel()". Away from the robotics domain, researchers have tackled unsupervised grammar induction from raw text inputs with the aim of replacing supervised techniques. Unsupervised techniques are favoured over supervised ones because of their ability to learn from unlabelled data. The process of annotating each sentence with a grammar tree is a labour intensive task that hinders the learning from large dataset. Also, labelled data is not necessarily available for all languages. Therefore, researches focused on tackling the problem of unsupervised grammar induction. For example, Ponvert *et al.* (2011) tackled the problem of unsupervised partial parsing, or unsupervised chunking of sentences using probabilistic finite-state method. Their work focused on learning how sentences can be parsed into smaller constituents by searching for repeated patterns in text. The work of Søgaard (2012) focused on tackling the problem of unsupervised dependency parsing without training. His system parses an input sentence using a combination of universal linguistic knowl-

[3]Interface: http://shrdlurn.sidaw.xyz

edge predefined initially, and the page rank algorithm. Further, Shain *et al.* (2016) presented a system that model the working memory limitations of young language learners in an unsupervised manner. The system learned the grammar rules from raw-text, using unsupervised hierarchical hidden Markov models. While unsupervised grammar induction techniques enable learning from unlabelled data, their performance is usually significantly worse than those of the supervised techniques. In this work, we present a novel technique capable of acquiring more accurate grammar rules from unlabelled data by mapping language to visual features extracted from video clips in a loosely-supervised manner.

Grammar induction techniques we also applied to model **visual inputs**. For example, Siskind *et al.* (2007) presented a system capable of learning the hierarchical structure of an image and its region using Probabilistic Context Free Grammar (PCFG). The hierarchical structure of the image regions is used in image analysis and classification. In their work, grammar rules were used to represent the spatial relations between the different regions in the image. Similarly, Socher *et al.* (2011) introduced a max-margin structure prediction architecture based on recursive neural networks. Their system was able to recover such structure both in images as well as sentences. However, it did not aim to learn the mapping between language and vision domains, but rather to show that the same technique can be applied successfully in both domains to learn the structure of images and sentences independently. Other researchers focused on using grammar to learn video structures. For example, Moore & Essa (2002) used PCFG rules to recognize multi-tasked activities from videos illustrating a Blackjack card game. However, the production rules describing all relations between the tracked events were manually-defined and not learned. Further, Yang *et al.* (2015) presented a system capable of learning cooking action plans from unconstrained video inputs. In their paper, the Viterbi probabilistic parser (1997) was used to represent different cooking actions in the form of a hierarchical and recursive tree structures.

Our work offers a novel loosely-supervised grammar induction approach from raw language and vision inputs. Our approach is developed to parse sentences into grammar trees with meaningful labels and probabilistic grammar rules. The learning of meaningful grammar rules is enabled by the use of both language and vision pairs as inputs, where grammar rules obtain their meanings from the vision domain. A detailed explanation of our approach is presented in

Chapter 5.

2.3 Perception in Robotics

Understanding how robots perceive the world and their own movements is essential for accomplishing navigation and manipulation tasks. In this work, we process images and videos acquired by cameras mounted on robots into predefined intermediate representations we call *visual features*. The values of these visual features are measured and accumulated in an incremental manner from recorded video clips. We then cluster these values into meaningful concepts we call *visual concepts*: which are abstractions of the visual feature spaces generated by the robot sensing modalities which carry a human-level meaning. For example, we consider the Red-Green-Blue (RGB) space as a *visual feature*, while the red colour as a *visual concept* in the RGB space. Ideally, we want our robots to be able to generate their own visual feature spaces, providing them with the ability to learn about any visual concept. However, this remains an ambition for the future.

We define a number of visual features and provide our robots with the ability to measure them from recorded video clips. The visual features selection was biased towards the available sensing modalities our robots have, and the nature of the objects present in the environment. The features are classified into five categories. First, object related features that include object colours, shapes, and locations. Second, human related features that include people's look (or faces) and their garments' colours. Third, spatial features that include relative directions and distances. Fourth, robot action features that include spatio-temporal graphs. Finally, human activity features that include Spatio-temporal relations and graphs. The visual features are presented in detail in Chapter 3.

The aim of our perceptual learning is to enable each robot to learn the unique visual concepts in its environment given only the definition of the visual feature spaces. For example, we provide one of our robots with the ability to detect people and extract features representing their faces, with the aim of enabling this robot to learn the different people that work/live in its environment. This reduces the amount of hard-coded initial knowledge significantly, and allows our robots to learn new concepts incrementally without the need for a professional programmer to predefine these visual concepts.

The following sections walk through the available literature in perceptual learning in robotics. We present state-of-the-art learning approaches in each of the five classes we define in this thesis, namely, object properties, people attributes, spatial relations, robot actions and human activities. We also compare our approach against state-of-the-art systems where applicable.

2.3.1 Learning object properties

Researchers at the American Academy of Pediatrics (1993) suggest that children's ability to recognize different colours, sizes, shapes and textures improves around 18 months. It will be a while longer before they can recognise basic shapes and colours, but most children can name at least one by the age of 36 months. We aim to enable our robots of learning about object properties such as object colours, shapes, etc., taking inspiration from how children learn about objects in their environment. We tackle this problem in a loosely-supervised manner, using a show-and-tell procedure; this is inspired by how children acquire knowledge of their everyday physical world by interacting with their parents.

A number of researchers have tackled the problem of learning object properties for robotic applications. One of the earliest works to tackle this problem in a robotic setup was the work of Pfeifer & Scheier (1997). They showed that the problem of object categorisation based on sizes was greatly simplified when the robot's own movements and interactions were utilized. In particular, a robot could grasp and lift small objects, push medium objects but not lift them, and do nothing with large objects. Through this interaction the robot was able to learn different object classes by interacting with them. Further, Roy *et al.* (1999) designed one of the earliest robotic systems capable of learning object shapes from static images. He used a simple background subtraction algorithm to detect objects along with various 2D contour features to learn object shapes. The system was deployed on a small robot arm which enabled it to learn different objects. Several robotic systems aiming to learn object properties were developed subsequently. For example, Rusu *et al.* (2008) presented a framework capable of acquiring location information of predefined objects in a room. Their system aimed at enabling robots to acquire object positions when deployed in new environments, and thus allowing robots to perform given tasks in new places. Furthermore, Griffith *et al.* (2009) proposed a framework that enables a robot to learn simple object categories such as identifying containers from non-containers. Their proposed approach is based on the principle that robots should learn object categories in their own sensorimotor experience. Their system was able to learn container/non-container categorization of objects by observing the motion behaviour that follows throwing a small block on top of the object. Similarly, the work of Kaboli *et al.* (2014) aimed at enabling a humanoid robot of learning object properties by interacting with them. Their robot was able to learn about textures and weights of different objects by sliding them against each other and measuring the forces involved in the process. Also, Sinapov *et al.* (2016) implemented a system capable of learning object properties such as weight, height, size, etc. Their robot learned the object properties by measuring them using a set of predefined actions such as 'pick up' to measure weight, 'drop' to measure sound and infer material type, 'grasp' to measure stiffness, etc. Their aim was to enable a robot to arrange objects based on their various properties, which is thought to be fundamental for human children to understand the property of numbers. However, their system was provided with the knowledge of robot actions (e.g. pick up, drop, etc.) needed to enable the learning of such properties.

In the works described above, the learning of object properties was enabled (made simpler) by using one of the following assumptions:

– the robot was presented with a single object in the scene to learn from.

– a teacher provided feedback to enable and correct the learning of properties.

– the learning was performed in batch mode (not incremental), where the robot was provided with a fixed-length dataset to learn from.

The main contribution we offer in the field of learning object properties is that we learn the properties in an incremental loosely-supervised manner, without teacher's supervision or feedback, with multiple objects present in the scene, and from real-world noisy data where objects are partially occluded and viewed from different angles. We also combine the learning of object properties with learning about other features such as spatial relations, robot actions, and at the same time learn the words and phrases in natural language used to describe the learned object properties. Hence, we build a framework the enables robots of bootstrapping their knowledge in language and vision.

2.3.2 Learning people attributes

To integrate in human environments, robots with collaborative/assistive human-oriented tasks should be enabled to continuously learn about the people who inhabit these environments. Researchers have tackled the problem of learning people's attribute using different techniques. For example, Berg (2004) presented a system that learns about names and faces from labelled news images. The images were processed to find the different faces, and then rectified to correct the face posture. The text (image labels) was processed to extract all names. For example, "*British director **Sam Mendes** and his partner actress **Kate Winslet** arrive at the London premiere of 'The Road to Perdition', September 18, 2002*". These names were used as features to improve the learning of faces. However, their learning approach was not incremental and required processing of all data when a new image is added to the dataset. Further, Fukui & Yamaguchi (2005) presented an approach to learn people's faces using multiple face patterns obtained from various views. By tackling the multi-view problem in face recognition, their system was able to improve the recognition results. However, it required supervised training of the faces. Recently, with the new development of deep learning techniques, researchers formulated the face recognition problem into a deep learning framework. For example, Parkhi *et al.* (2015) implemented a Convolutional Neural Network (CNN) to classify each individual that appeared in a large dataset. The network was trained to identify a total of 2622 unique faces that appeared in their 2.6 million images dataset. Their CNN implementation achieved state-of-the-art results in face recognition. However, the supervised training of CNNs, and the need for annotated data to learn from makes deep learning approaches less desirable for our incremental and loosely-supervised framework for autonomous robots.

Not many researchers focused on learning people's attributes using a mobile robot collecting noisy, real-world and partially occluded data. Also, the works described above address this problem mostly in a supervised and not incremental manner. We believe that the nature of the learning should be incremental, as new people may join the environment where the robot operates. Therefore, we designed our system to learn about people in an incremental loosely-supervised manner, and without the need for a teacher to correct the learning. We also learn the names of these people using our language grounding framework.

2.3.3 Learning spatial relations

The recent advancements in robotics manipulation research are enabling robots of performing a wide range of tasks, and allow them to move from carefully engineered to open and unknown environments. This raised the need for a concise representation of a widely varying world. For example, a robot may encounter objects in a wide variety of configurations. In order for the robot to be able to manipulate these objects, it needs to understand the qualitative relations between them. Qualitative Spatial Representation (QSR) offers a suitable answer to this problem. QSRs also align well with how humans represent and describe the world, which makes this representation more suitable for robots designed to interact with humans and learn from them.

Researchers have defined a large number of qualitative relations. For example, the survey paper presented by Chen *et al.* (2013) shows that there exist numerous types of QSRs. Also, an open-source library that encodes various types of QSRs was presented by Gatsoulis *et al.* (2016a). For brevity, we will mention only the ones used in this thesis. Three QSRs are used to represent human activities. First, the Qualitative Distance Calculus (QDC) presented by Clementini *et al.* (1997) expresses the qualitative Euclidean distance between two points based on predefined distance thresholds. A set of QDC relations can be used to localise an object or a person with respect to reference landmarks. By observing changes in QDC relations, we can model human activities. For example, we can use QDC to model a drinking action. First, a tea cup is 'far' from the person's face. Then, it's 'touching' the face. Then, it's back to being 'far', These changes in relations are used to model human activities. Second, the Ternary Point Configuration Calculus (TPCC) presented by Moratz and Ragni (2008) describes in a qualitative way the spatial arrangement of a point relative to two others. That is, it describes the *referent*'s position relative to the plane created by connecting the two other points (which are referred to as *relatum* and *origin*). Relations in TPCC are triples of $\langle$ { front, back }, { left, right, straight }, { distant, close } $\rangle$. Third, the Qualitative Trajectory Calculus (QTC) presented by Delafontaine *et al.* (2011) represents the relative motion of two points with respect to the reference line connecting them, and is computed over consecutive timepoints. For two objects o_1, o_2, it defines the following three relations (objects are abstracted to their centroids when computing QSRs). : $\{o_1$ is moving towards o_2 (symbol $-$), o_1 is moving away from o_2 $(+)$, o_1 is neither moving towards nor away from o_2 $(0)\}$.

Researchers have also tackled the problem of learning QSRs from observations. For example, Galata *et al.* (2002) presented an unsupervised approach to learn spatial relationships between moving objects. Using their approach, a system can learn to cluster distances and directions of motion into a set of QSRs from unlabelled traffic videos of moving vehicles. The system was able to learn different motion patterns between pairs of cars, and use these relations to represent the different driving activities that occur in the videos such as overtaking, passing, etc. Similarly, Rosman & Ramamoorthy (2011) presented a system capable of learning distance and direction based QSRs but in a supervised manner. Their system takes as inputs segmented 3D point-clouds (i.e. the unique objects were manually defined) and extracts quantitative measurements of distances and directions at the points of contact between objects, which are clustered into different relations using k-means clustering presented by MacQueen *et al.* (1967). These relations are then used to represent different spatial configurations between objects, such as representing a tower of block. Also, Sjöö & Jensfelt (2011) presented a system capable of learning functional spatial relations from a simulated world. The simulation is used to produce static scenes with random objects placed in various configurations. A physics engine was also used to simulate the interactions between these objects. Their system was capable of learning five functional distinctions: effective support, support force, location control, confinement and protection. However, their experiments were limited to simulated scenes with noise-free and fully observable measurements. The work of Behera *et al.* (2012) presented a system that learns distance and velocity qualitative relations. Their work aimed at monitoring the workflow of assembling various products such as 'hammering nails' and 'driving screws'. The learning of QSRs was enabled by quantising distance and velocity measurements into a finite number of states using a Hidden Markov Model (HMM). However, the objects types were known to the system before hand by using Vicon markers on all key objects, including both wrists of the participants. Further, Kunze *et al.* (2014) presented a system that bootstraps robot's knowledge in QSRs in an office environment. The system learns the different relations (directions and distances) by observing the relations between different objects on desks. These relations are used to facilitate the object search task by narrowing the search space. For example, by knowing that a computer mouse (a small object) is often placed close and to the right of a keyboard (a bigger and easier to find object). Then the problem of searching for the computer mouse becomes easier. Similarly, Boularias *et al.* (2016) implemented a system that learns QSR in directions and distances but with the aim of improving robot navigation. Their system aims to enable a mobile robot of understanding commands such as "*Navigate around the building to the car* ***left*** *of the fire hydrant and* ***near*** *the tree*". To understand such commands, the robots need to have a working knowledge of spatial relations. Their system was able to learn different relations such as *"left", "right", "front", "near", etc.* However, they assumed the objects were known beforehand to enable the learning of spatial relations.

In this work, we present a system capable of learning QSRs from noisy real-world data, along with learning about object properties and robot actions. We also learn to ground words from natural language descriptions to describe these learned relations, which enable our robot to understand natural language commands with spatial information such as "*place the ball on top of the red block*", or "*place the apple in the bowl*".

2.3.4 Learning robot actions

Autonomous robots are becoming an important part of our society, whether an exploration rover on Mars or service robot for the home. In order for these robots to operate successfully in any environment, they need the knowledge of how to perform the different requested tasks. Such knowledge can be provided by an expert. But, as robots become more available to non-expert users, they will need to learn the different actions by observing and imitating their users.

In the literature, several approaches were developed to enable the learning robot actions from non-expert users. For example, Calinon & Billard (2007) implemented a system that enables robots of learning gestures by imitation. A user performs a demonstration of a gesture while wearing motion sensors recording his upper-body movements. These recordings are then used to learn each gesture using a Gaussian mixture model to model the motion. However, their system was only able to learn gestures which can not be generalised to actions manipulating objects at different locations. Similarly, Pastor *et al.* (2009) implemented a system capable of learning robotic motor skills from human demonstration. Their system learns movements by learning a non-linear differential equation to represent and reproduce each movement. The differential equations can be generalised to adapt to different locations by adapting a start and a goal parameter in the equation to the desired position values of a movement. However, their learned actions lacks the notion of object manipulations and focuses on primitive movements only. Further, Ramirez-Amaro *et al.* (2015) presented a method that allows transferring skills to humanoid robots based on observations of human activities. The robots obtain a higher-level understanding of a demonstrator's behaviour using semantic representations. However, they simplified the vision problem by using the ArUco library to detect the AR marker on each object.

Our work is focused on learning actions in robotic manipulation domain. The learning of actions is enabled by the use of spatio-temporal graphs (Alomari et al. (2017b)). The use of spatio-temporal graphs enables our robots to learn relatively complex activities such as 'pick up', 'move', 'push', etc. by modelling the interactions between the arm and the objects, rather than modelling the motion of the arm itself. This enables the robots to generalise to new objects placed at different locations. A detailed explanation on how we learn robot action concepts is presented in Chapter 3.

2.3.5 Learning human activities

A key factor for the success of autonomous intelligent robots, deployed in human populated environments, is their ability to understand human activities. This allows for safer and more effective interaction with humans. Researcher have tackled the learning of human activities using supervised techniques. For example, Dubba *et al.* (2012) presented a system capable of learning activity models in a supervised manner using inductive logic programming technique. The system was able to learn from videos of an airport apron where events such as 'loading', 'unloading', 'jet-bridge parking', etc. took place. Similarly, Behera *et al.* (2012) implemented a supervised system that learns to recognise the workflow of assembling products. Their system used learned spatial relations to represent the interactions between hands and objects in each scene. An HMM was then trained to recognise these activities. Further, the work of Tayyub *et al.* (2014) focused on learning and recognising complex human high-level activities in a supervised manner by training an SVM. Their method was based on using both qualitative and quantitative spatio-temporal features to capture the person-object interactions in each scene.

Some researchers tackled the problem of learning human activities in an unsupervised manner. This allows robot to learn from large datasets, as no human annotation or supervision is needed. For example, Sridhar *et al.* (2008) used spatio-temporal graphs to represent the time series data for unsupervised learning of event classes from video clips. The activities were modelled using a set of predefined qualitative spatial relations and temporal relations using Allen's intervals (1983). Both spatial and temporal relations were combined into graph structures called spatio-temporal graphs. Their system mines these graphs and use them as features to represent and learn the different activities. Similarly, Duckworth *et al.* (2016b) presented a system for unsupervised learning of human motion patterns in an office environment. The data was collected using a mobile robot patrolling an office environment for over one month. Their system used spatio-temporal graphs, similar to that presented by Sridhar *et al.* (2008), which were used as features to learn the different motion patterns using k-means clustering. Further, Duckworth *et al.* (2017) extended their unsupervised approach to model more complex human activities. They collected data that included skeleton tracking of humans in a kitchen environment. The data was then used to learn complex activities such as 'making tea', 'microwaving food', etc. in an unsupervised manner. The learning of activity models was achieved using Latent Drichlet Allocation (LDA) technique. However, the learning was enabled by using a set of manually annotated objects on the map such as 'fridge', 'microwave', etc.

To learn human activities, we follow the work of Duckworth *et al.* (2017), which I co-authored. We learn human activity models in an unsupervised manner using LDA technique. Also, we extend the work by learning from a set of discovered objects rather than manually defined ones, along with learning words in language used to describe the learned activities using our language grounding framework as presented in our work Alomari *et al.* (2017a).

2.4 Summary

We tackle a number of research problems in an incremental and loosely-supervised manner. First, can a robot acquire human-level visual concepts from its sensory-motor experience? Second, can a robot learn the meaning of words by grounding them into the learned visual concepts? Finally, can a robot learn about language grammar in a loosely-supervised manner from language and vision data?

As presented in this chapter, researchers have tackled these questions using various techniques and assumptions. The key contribution this work offers is that it aims to answer all three questions concurrently. Our learning framework aims to learn about the visual world and natural language at the same time without assuming one is provided to learn about the other. For example, some researchers assumed their robots know about natural language and are capable of having a conversations with a teacher to enable the learning of visual concepts, such as learning about new objects or actions. Others assumed their robots know about the visual world, and are capable of recognising objects or rooms in an environment to enable the learning of natural language, such as learning to understand natural language commands and grounding words to vision.

We aim to bootstrap a robot's knowledge in both language and vision simultaneously. We show that a robot can start with little hard-coded knowledge and can still learn about language and vision. The following chapters walk through the details of how we achieve this goal. We start by describing the learning of the vision domain in the next chapter.

Chapter 3

Visual Concepts

In this section we introduce our notion of *visual concepts*: abstractions of the feature spaces generated by the robot sensing modalities which carry a human-level meaning. For example, concepts might include a colour represented as a cluster of values in the HSL colour space (Hue-Saturation-Lightness), or an object represented as a cluster of points in a 3D point cloud, or a complex human activity represented as a probability distribution over spatio-temporal graphs. We present in the following sections the robots, sensors and feature spaces used along with the unsupervised methods employed to generate such concepts. Note that our visual concept extraction framework does not rely on any particular robot or any specific sensors; rather it is flexible to what the modalities of the robot can support.

3.1 Robots and Sensors

Four different robots are used to validate our learning approach of language and vision: (*i*) A Scitos A5 mobile robot from MetraLabs (2016) (named LUCIE) running Robotics Operating System (ROS) Indigo (2009) and the full STRANDS system (2016); (*ii*) A Baxter robot from Rethink-Robotics (2013) (named LUCAS) that has two arms and two fingered grippers; (*iii*) A custom made mobile manipulator by Sinapov *et al.* (2016) that uses the Segway Robotic Mobility platform (2004) and a 6-DOF Kinova Mico arm (2014) with a two fingered gripper as its end effector; and (*iv*) A simulated 3-DOF robotic arm with a two fingered gripper in a chess-board simulation environment that I developed using the Python programming language presented by Alomari *et al.* (2016a). The four robots are shown in Figure 3.1.

The robots are equipped with at least one sensor that allows mapping of the robots' environments. For example, LUCIE (the mobile robot) has two *Xtion* PRO LIVE sensors (2014); one over-head and one chest-mounted, while the Kinova mobile manipulator has one fitted on its base. The *Xtion*s allow safe navigation and collection of 640×480 RGB video streams in addition to depth point clouds of the environment. On the other hand, LUCAS (the Baxter robot) is equipped with one chest mounted Kinect2 sensor (2015) that allows collecting of 1920×1080 RGB video streams in addition to high resolution depth point clouds of the environment from the robot's perspective. These sensors are used to collect the data needed to learn the visual concepts. Note that the 4-DOF simulated robot arm has no sensors and is assumed to have access to full observations from the environment.

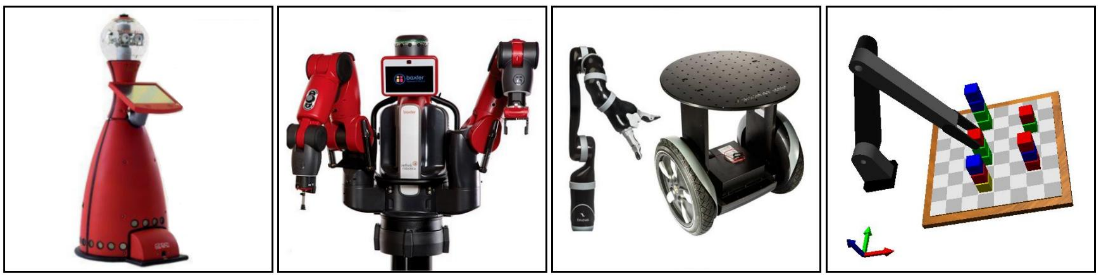


Figure 3.1: From left to right, Scitos A5 mobile robot (LUCIE), Baxter robot (LUCAS), custom built mobile manipulator, and the 4-DOF robot arm in a chess-board simulation.

3.2 Low-Level Processing of Input Data

The robots are used to collect instances (short video clips) of the environment, where each instance contains at least one action performed by either a robot or a person, e.g. a person printing or making tea, or a robot moving an object. Each recorded video clip is processed to detect and track humans and objects in the scene. This helps the robot in identifying and focusing on the interesting concepts in the environment and enables the learning of such concepts. The details of how the robots detect and track both humans and objects in each recorded video clip are given in the following two sections.

3.2.1 Human pose estimation

The mobile robot LUCIE detects and tracks humans as they pass within the field of view of its head-mounted RGB-D sensor. We define a human pose as the estimated 3D position of the person's 15 body joint locations at a single frame in a video clip. The 15 body joints are the *head, neck, torso, shoulders, elbows, hands, hips, knees* and *feet.* For each body joint j, its (xyz) Cartesian coordinates are inferred, and a *human pose estimate* comprises of 15 such joints $J = [j_1, j_2, \ldots, j_{15}]$. To estimate the human pose, a real-time depth-only tracker built on OpenNI (2016) is used along with a post-processing state-of-the-art human pose estimation technique that uses convolutional pose machines (CPM) by Wei *et al.* (2016). For each human detected by the robot, a sequence of human pose estimates over a time series of frames is acquired, e.g. Figure 3.2 shows two pose estimates for a detected person at two different times in a recorded video clip.

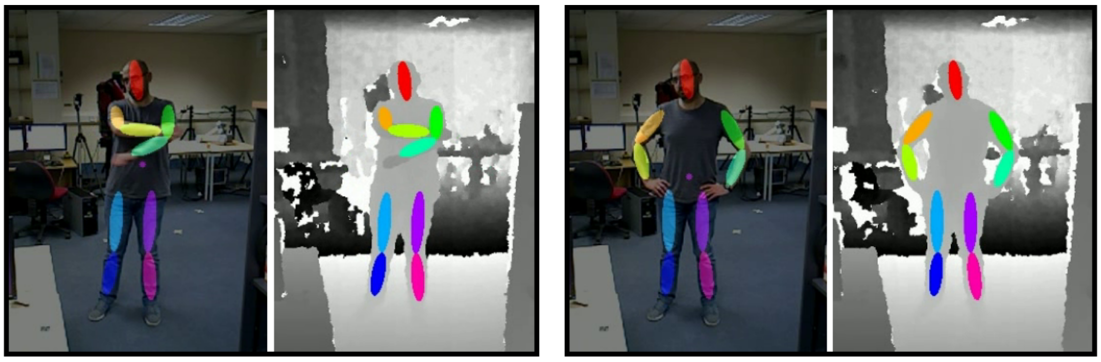


Figure 3.2: Examples of detected human poses, using inputs from the head-sensor of the mobile robot LUCIE. The 15 body joints are comprised of the head, neck, torso, shoulders, elbows, hands, hips, knees and feet.

3.2.2 Object detection and tracking

The robot constructs a 3D model of its environment by fusing RGB-D images into *surfels*: surface elements used as rendering primitives introduced by Pfister *et al.* (2000), from which the robot generates segments of "objects of interest". As demonstrated by Schoeler *et al.* (2015), an unsupervised segmentation algorithm grounded in the convexity of common human objects can achieve state-of-the-art performance in extracting semantically meaningful object segments.

We use the method presented by Bore *et al.* (2017) to segment the scene, which first splits the scene into a collection of *supervoxels*: a polyhedral part of a three-dimensional digital image introduced by Papon *et al.* (2013), over which an adjacency graph is formed. Then, weights are assigned to the edges based on local convexity of the point cloud and colour differences between segments. Finally, to segment the point cloud, iterative graph cuts are performed to separate parts with concave boundaries and/or large colour differences. This results in a collection of point cloud segments or objects of interest as illustrated in Figure 3.3.

In the mobile robot environment, it is important to concentrate attention on the objects that are part of the observed human activities. First, walls, floors and ceilings are removed from the list of objects of interest using a threshold on size and height. Second, the trajectories in 3D space of people in the environment are analysed to extract the locations where people stop more frequently. The objects are scored according to their proximity to people's hands in these locations. The highest scoring objects are considered as the only objects in the environment.

In the manipulator robot environment, to concentrate attention on the objects that are part of the observed manipulation activities, a "table-top" object detection technique by Muja and Ciocarlie (2013) is used to drive the attention of the robot to the graspable objects placed on a table within the robot's reach. Once an object is detected in a video clip, the location of this object is tracked across all remaining frames using a six dimensional particle filter presented by Klank *et al.* (2009). The six dimensions are the three x, y, z location and the three r, g, b colour values of each pixel in the object segment.

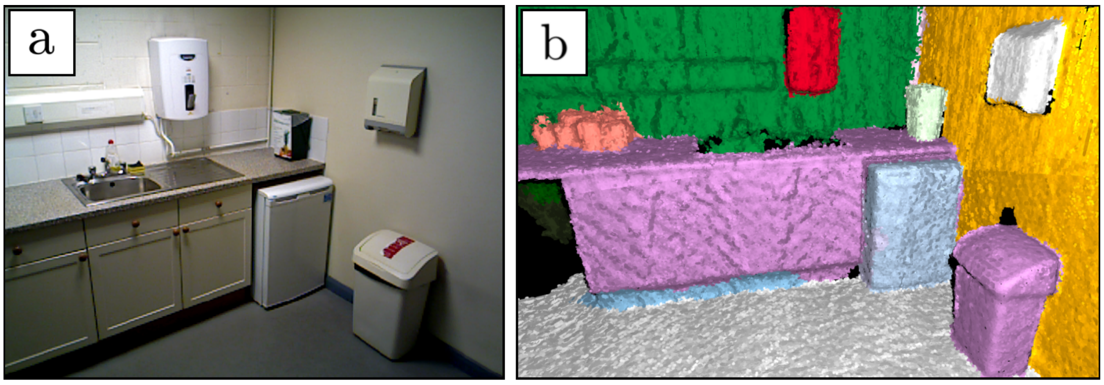


Figure 3.3: Processing of 3D data on the robot. The environment observations are fused into a 3D map and segmented. (a) RGB image of the scene, (b) segmented surfel map, or the objects of potential interest.

3.3 Concept Extraction

Concepts are learned automatically by clustering the low-level sensory input of each of the sensor modalities of the robot after an appropriate encoding. This clustering operation results in a collection of classes that are candidate concepts within each feature space. Because we assume no pre-knowledge of the structure of the sensor feature spaces, we employ probabilistic modelling techniques to each feature space independently to elicit meaningful classes that are supported by the observed data. These classes are used as our candidate visual concepts and will later be used to learn the mapping between natural language and vision.

We differentiate between two kinds of visual concepts, (i) *simple concepts*: ones that can be detected in a single observation. For example, objects are simple concepts that can be segmented from 3D point clouds using geometrical and textural cues as per recent literature by Bore *et al.* (2017). Similarly, concepts like colours can be represented as Gaussian components in a Gaussian Mixture Model over the HSL space. On the other hand, (ii) *complex concepts*: ones that manifest over longer sequences of observations. For instance, temporally-extended human activities and robot actions are examples of complex concepts. For these, a more elaborate encoding and more sophisticated clustering mechanism are needed as per recent literature by Duckworth *et al.* (2016a; 2017). For human activities, the robot first abstracts each observed human pose sequence using a qualitative representation and obtains clusters using a hierarchical Bayesian model, Latent Dirichlet Allocation (LDA) presented by Blei *et al.* (2003). It translates the detected pose sequence into a relatively small number of logical spatial relations that can be used to qualitatively describe the interactions taking place between the person and the objects in the environment. The topics recovered from this process are considered human activity concepts which the robot learns and grounds to words in natural language.

Our visual concept extraction framework is demonstrated by extracting five kinds of concepts from raw data obtained by deploying four robots in different environments. The five visual concepts are; (i) *Human related concepts*: ones used to represent peoples faces and the appearance of their garments, (ii) *Object related concepts*: ones used to describe object properties such as object's shape, colour and location on a table, (iii) *Spatial relation concepts*: ones used to represent spatial relations between pairs of objects like the relative distance and direction between them, (*iv*) *Robot action concepts*: complex concepts (i.e. temporally extended ones that require more elaborate encoding) used to represent the actions such as object lifting, pushing and moving performed by the different robotic manipulators used in this thesis, and finally (*v*) *Human activity concepts*: complex concepts used to represent various human activities from walking through a door to preparing a meal observed/recorded by the mobile robot LUCIE. In the following sections, we introduce each of the feature spaces used in this thesis and demonstrate how robots cluster observations in each of them to obtain candidate visual concepts that model their environments.

3.3.1 Human related concepts

Only LUCIE (the mobile robot) was used to observe and learn from humans. By deploying LUCIE in a human populated environment, we aim to learn about the people that live or work there. This is achieved by processing each detected person to extract facial and colour features. The aim is to use the facial features to learn people's names by finding the different (unique) faces in an environment, and to use the colours of their clothes to learn people's garment description.

Faces

Recent experiments in child development by Turati *et al.* (2006) have shown that even one to three day old babies are capable of distinguishing between known and unknown faces. So how hard could it be for a mobile robot fully equipped with RGB-D sensors? One of the key challenges in this task is the variation in multiple viewpoint observations obtained for the same person by the mobile robot. Fukui & Yamaguchi (2005) tackled the multiple viewpoint face recognition problem using a supervised technique, while Berg (2004) learned about names and faces from labelled news data in a supervised setting. We tackle the learning of names and faces in an incremental and loosely-supervised setting, from a noisy real-world dataset obtained by a mobile robot patrolling in an office environment where faces can be partially occluded and viewed from different angles. To learn and recognise people's faces, a small patch around the location of the head joint is automatically cropped from the RGB visual feed for every person detection, the head joint is estimated using the human pose convolutional pose machine in 3.2.1. The presence of a face in the cropped images is detected using a cascade of boosted classifiers with Haar features presented by Lienhart & Maydt (2002) along with the OpenCV generic face model. Then, the Eigenvalues for the n_f most prominent 'Eigenfaces' are extracted as presented by Turk and Pentland (1991). This transforms a face instance into a much-smaller n_f-dimensional data point. Then, we fit a Gaussian mixture model (GMM) in that space with an optimal number of components selected in an unsupervised manner using the Bayesian Information Criterion (BIC) by Posada and Buckley (2004). The resulting Gaussian components are used as candidate visual concepts to represent different people that the robot encounters in its environment, which facilitate the grounding and learning of language and vision (i.e. learning to recognise people and their names). Examples of face clusters found by LUCIE are shown in Figure 3.4.

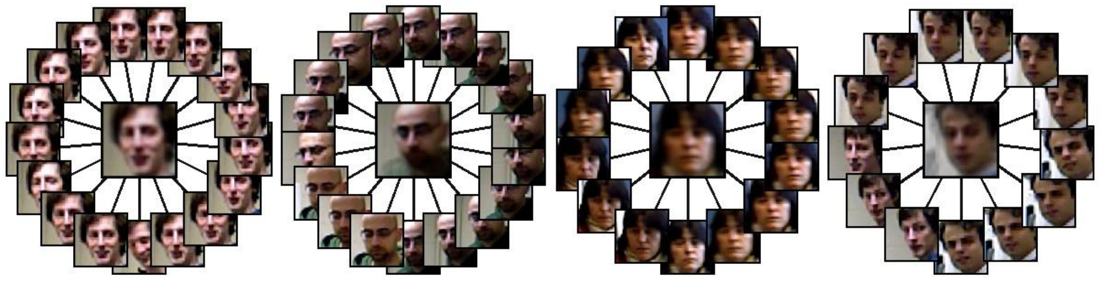


Figure 3.4: Examples of four face clusters found automatically using GMM and BIC techniques, with the averaged (mean) face shown in the center of each.

Colours

The robot learns about people's description by observing the colours of their garments. The values of the upper and lower garment's colours of each person detection is extracted and then clustered into candidate visual concepts. The colours of the upper and lower garments are extracted from the visual feed using the human pose estimate 3.2.1, where the colour of the upper garment is estimated by the average of sampled pixel colours from the triangle of the shoulders and torso, and the colour of the lower garment is sampled from the triangle between the torso and knees, as shown in Figure 3.5 (left). The clustering is achieved by fitting a Gaussian mixture model. The number of Gaussian components is selected automatically using the BIC. The extracted colours are projected into a single Hue-Saturation-Lightness (HSL) feature space where they are clustered. Using the HSL feature space as opposed to using RGB increases the robustness of colour recognition under varying lighting conditions. Examples of six colour clusters extracted by our mobile robot LUCIE are shown in Figure 3.5 (right).

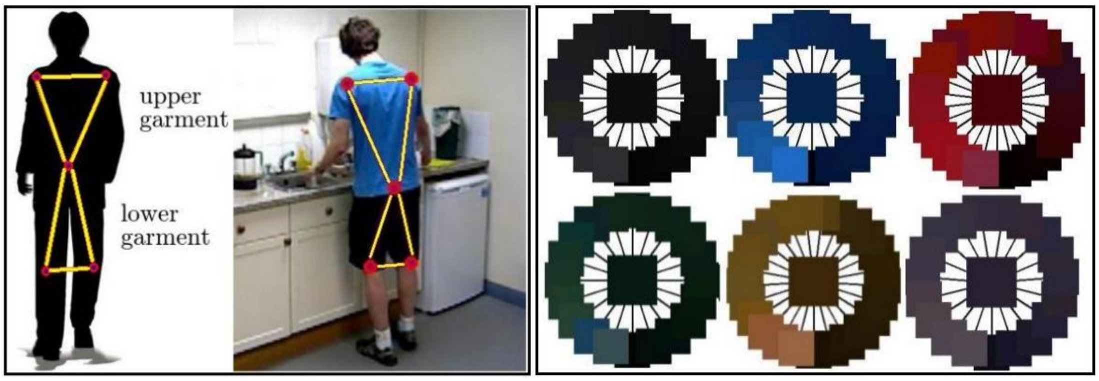


Figure 3.5: Left: defining upper and lower garments using human pose estimate 3.2.1 (Best viewed in colour). Right: examples of six different colour clusters with the averaged (mean) colour shown in the center of each cluster.

3.3.2 Object related concepts

A number of studies by the American Academy of Pediatrics (1993) suggest that the human child's ability to recognize different colours improves around 18 months, the same time s/he begins to notice differences and similarities in size, shape, and texture of different objects in the environment. It will be a while longer before s/he can recognise the basic shapes and colours, but most children can name at least one by the age of 36 months. We tackle the learning of object properties (e.g. shape, colour, and location on a table) and the words describing these properties in natural language, while keeping the cognitive plausibility of our learning framework in mind. A number of studies on language acquisition in human infants such as Piaget (1954), Berlin and Kay (1969), Rosch and Mervis (1977) and Bowerman (1996) have suggested that children come to develop considerable cognitive understanding about the objects they encounter in the prelinguist period of their lives. In other words, their studies suggest that we come to learn about object properties before we learn the words used to describe them in natural language. It should be noted that this work does not claim to be modelling how humans learn about objects in their environments, but rather it is inspired by how we learn about objects, and it also mimics certain aspects of this complex learning process. Specifically, we assume that our robot has no pre-given knowledge about the number of unique concepts in each feature space or the words used to describe them in natural language (e.g. the basic colours or what they are called in English or any other language), and we also assume that the robot learns about the object properties before it learns the words used to describe them as suggested by the language acquisition studies in human infants. In the following sections, we describe in detail the unsupervised techniques used by the robot to acquire object-related visual concepts from raw visual data. For each detected object, we aim to learn about its properties (shape, colour, and location on a table). This is achieved by clustering the values in these continuous feature spaces into a number of candidate visual concepts. This set of features is not intended to be exhaustive, but rather to demonstrate our approach. Other features could be easily added, such as size and texture of objects.

Shapes

To learn and recognise the different objects found in the environment, the robot examines the "objects of interest" extracted from the 3D model of the environment using the unsupervised segmentation techniques described in §Objects detection and tracking 3.2.2. Each segmented object is processed to extract features describing/representing its shape. We use the fast point feature histogram (FPFH) presented by Rusu (2009) for this purpose. The FPFH is a multi-dimensional histogram of features which describe the local geometry around a point p in a 3D point cloud, that is scale and view invariant and copes very well with different sampling densities or noise levels presented in the recorded point cloud. Examples of FPFH values for four objects in a point cloud are shown in Figure 3.6 (left). FPFH is used to generate a 33 bin histogram of features for every detected object of interest, the bins are basically counts that measure the angles between the normal vector of point p and the normals of its k nearest neighbouring points. This technique is shown to work well in representing various table-top objects in the literature by Rusu (2010) and Rusu *et al.* (2010). Once the FPFH values are computed for all objects in the scene, they are projected into one feature space where each value represent a single datapoint in that feature space. Then, we fit a Gaussian mixture model in that space with an optimal number of components selected unsupervised using a Bayesian Information Criterion (BIC). The resulting Gaussian components are used as candidate concepts to represent unique shapes/objects in the environment. Examples of such clusters from the Baxter robot environment are shown in Figure 3.6 (right).

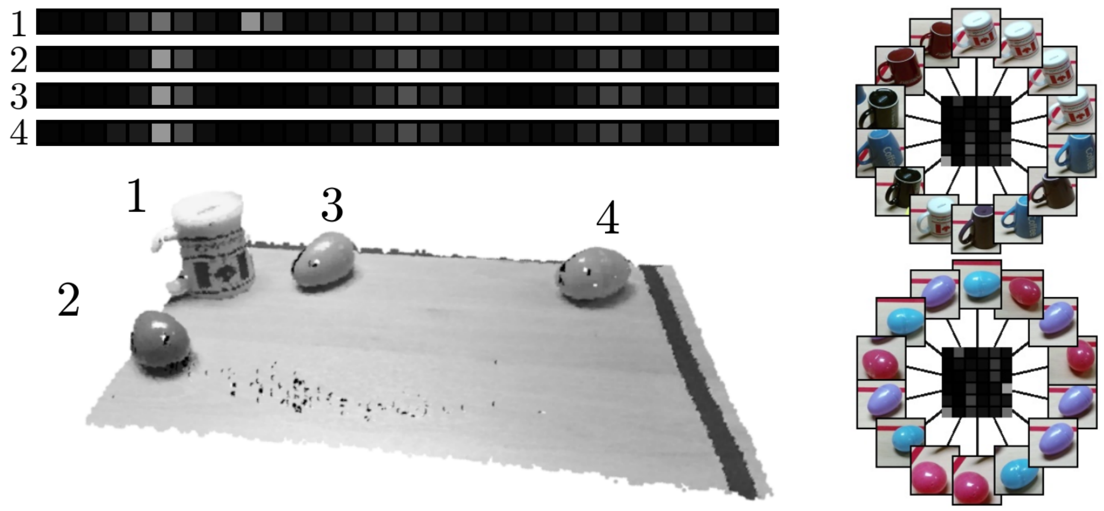


Figure 3.6: Left: Examples of Fast Point Feature Histograms for four objects in a point cloud. Right: Examples of two different object clusters with the averaged (mean) values of each of the 33 bins of FPFH shown in the center of each cluster.

Colours

We aim to teach our robot about the basic object colours observed in its environment, similar to how we learn about human garment colours in 3.3.1. The robot examines the segmented objects detected in the environment (the "objects of interest") to extract their colour values. For each detected object in a scene (a recorded video clip), we measure the HSL (Hue-Saturation-Lightness) values of every pixel in the object segment at every frame. The values extracted from each video clip are projected into a single HSL feature space and clustered using GMM to obtain the unique concepts, i.e. unique colours. The number of components is selected unsupervised using BIC technique, similar to how we learn garment colours in 3.3.1. The only difference between object and garment colours, is the way we measure the colour values. In object, we measure the colour value for every pixel in the object segment, while for garments, we define an upper and lower triangles as shown in Figure 3.5.

Locations

To learn and recognise the different canonical locations on a table (e.g. centre of the table, top right corner), we measure the x,y,z location of the centre of each object segment. The measurements are taken at every frame in a video clip, and the measured values are projected into a single x,y,z feature space, where the location values are clustered into Gaussian components (unique locations on a table) using GMM. The number of components is selected unsupervised using a BIC.

3.3.3 Spatial concepts

Human children as early as three years old are capable of distinguishing a number of basic spatial relations such as telling "left" from "right", and "in" from "on", and most children develop a firm grasp of such spatial relations by the age of seven or eight. As they get older, they can do more complex tasks that involve more complex spatial relations according to Bowerman (1996). We take a cognitive approach to teach the robot about a number of simple pair-wise spatial relations by first extracting the spatial concepts from raw visual data, then grounding words to these concepts as suggested by language development in human infants literature such as Piaget (1954) and Bowerman (1996). For every pair of detected objects in a scene, two pair-wise spatial relations are computed: Euclidean distance, and relative direction (azimuth and altitude angles). The robot is assumed to start with no pre-given knowledge in any of these feature spaces, e.g. the number of concepts in any of these relations, or the language used to describe them. In the following sections, we describe how the robot extracts and clusters the values of these relations from raw visual data.

Euclidean distance

To teach the robot about relative distances, for example *near* and *far*, the Euclidean distances between every pair of detected objects (the objects of interest from 3.2.2) in the scene are extracted at every frame, $distance : object \times object \rightarrow R$; $distance(o_1,o_2)$ gives the Euclidean distance between the centroids of object o_1 and object o_2. Once the values are measured between all pairs of objects at every frame in the scene, they are clustered into Gaussian components using GMM, and BIC to select the optimal number of components. These Gaussian components represent the candidate spatial concepts in the distance feature space, and are used along with other extracted concepts to teach the robot about language and vision. We limit the learning to simple distances as opposed to learning comparatives and superlatives too, like (*further*, *furthest*), (*nearer*, *nearest*), etc. as learning such concepts requires a more complex notion of logic that we do not assume is innately provided to our robot.

Relative direction

To teach our robot about relative directions, e.g. *to the right of* and *on top of*, we use the horizontal coordinate system to measure the azimuth and altitude (or elevation) angles between every pair of detected objects in the recorded scenes at every frame, $direction : object \times object \rightarrow [0, 360) \times [0, 360)$; $dirction(o_1,o_2)$ gives the azimuth and altitude angles from the centroid of object o_1 to the centroid of object o_2 as shown in Figure 3.7 (left). This measurement is applied on all pairs of objects in the scene. The observation plane from which the altitude angle is measured is assumed to be the table plane, and the north direction from which the azimuth angle is measured is assumed to be the robot's heading as shown in Figure 3.7 (right). Once the values are measured between all pairs of objects, they are clustered into Gaussian components by fitting a GMM, and BIC to select the optimal number of components. We assume that the computation of the azimuth and altitude angles is dependent on the point of view of the observer (the robot) as we do not assume the individual objects have a main or principal axis or a front face to compute these angles from. Also, we limit the learning to pair-wise simple directions as opposed to learning superlatives too, such as *rightmost*, *leftmost*, etc. as learning these concepts require a more complex notion of logic.

3.3.4 Robot actions

We are interested in learning about the concepts in the robot's environment, and also about concepts related to the robot itself. Our robot is assumed to start with no given knowledge regarding what it can or can't do and what is worth doing. For example, the robot does not know if pushing an object is a useful action that is worth learning. To learn such concepts, i.e. robot-

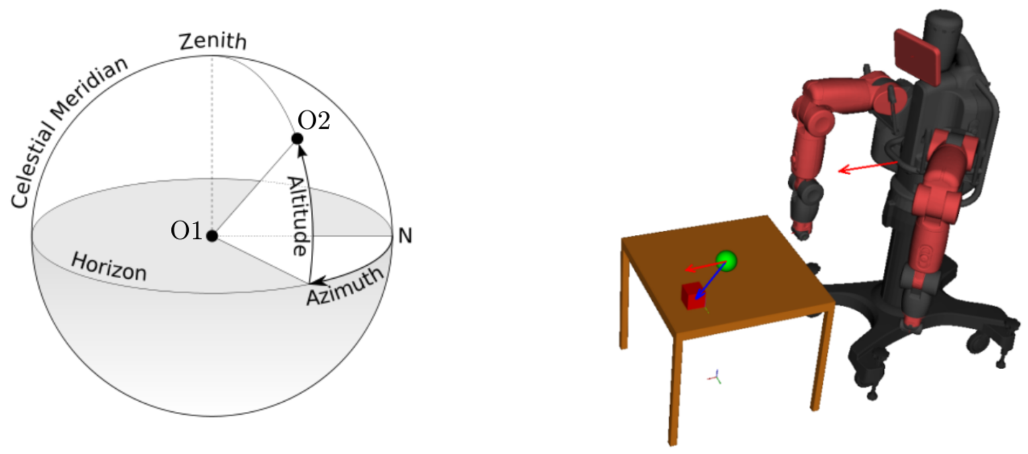


Figure 3.7: Left: The horizontal coordinate system uses the azimuth and altitude angles to measure the relative direction between two points in 3D space. Right: Utilizing the horizontal coordinate system to measure the relative directions between pairs of objects, the red arrow defines the "north" of the scene from which the azimuth angle is measured, while the table defines the plane to measure the altitude angle.

action concepts in a table-top setup, the robot is controlled by volunteers to demonstrate how to perform different table-top tasks in a loosely-supervised manner. The term loosely supervised refers to the kind of learning that requires the videos and sentences to be temporally aligned beforehand, which is more suitable for teaching infants about basic concepts such as colours or shapes. A fully unsupervised system would be able to learn from longer non-segmented videos and documents (i.e. be able to temporally segment and align long videos and documents), which remains an ambition for the future. The loosely-supervised teaching of robot-action concepts happens in the following way; if we ask the robot to move object A into object B, a volunteer would drive the robot arm using a joystick to perform this action while the robot records the changes in the environment. The joystick controls the velocity of the robot arm[1]. Using the recorded videos, the robot learns about the different actions using three processes: First, encoding the visual world into a number of predicates using the extracted visual concepts; Second, abstracting the changes in the visual world using "spatio temporal graphs"; Third, mining these graphs to obtain sub-graphs that represent the different objects, relations and actions. In the following sections, the learning of robotic actions is explained by describing each of these three steps in detail.

[1]Code and details are available in the github repository https://github.com/OMARI1988/baxter_pykdl

Visual world encoding

In order to teach the robot about actions, we first have to represent the objects and relations that are involved in each action. This is enabled using the extracted object concepts (colours, shapes and locations) and spatial concepts (relative directions and distances) shown in sections 3.3.2 and 3.3.3 respectively. Each video clip is processed separately to extract the unique spatial and object related concepts, which are used to represent the state of each object and spatial relation at every frame in the clip. The representation is made using a collection of predicates of the form: $\textit{object-concept}(\textit{object})$ for object properties; and $\textit{spatial-concept}(\textit{object}_1, \textit{object}_2)$ for spatial relations. To extract these predicates, first, each detected object is assigned a unique number (an $id = 1, \ldots, m$) while the robot gripper is assigned a unique $id = G_R$, and each visual concept is assigned an internal symbol, e.g. the cluster representing the red colour is $\textit{colour}_1$, the cluster of the cube shape is $\textit{shape}_1$, the cluster of the far distances is $\textit{distance}_1$[2], etc. Each internal symbol holds as a predicate on the identity of the object that has that sensory value in that dimension, which is decided using the Mahalanobis distance introduced by Mahalanobis (1936). The Mahalanobis distance d_M shown in Equation 3.1 measures the distance in an n_d dimensional feature space between an observation $X = \{x_1, \ldots, x_d\}$ and a multi-variant Gaussian distribution G with a mean $\mu = \{\mu_1, \ldots, \mu_d\}$ and an $d \times d$ standard deviation matrix Σ. The Mahalanobis distance measures how many standard deviations away X is from the mean of G. The distance is equal to zero if X is at the mean of G, and grows as X moves away from the mean.

$$d_M(X) = \sqrt{(X - \mu)^T \Sigma^{-1} (X - \mu)} \tag{3.1}$$

The Mahalanobis distance provides an indication of whether the observation X belongs to the distribution G or not. The observation X in our case is a measured value of an object property (colour, shape, location) or a spatial relation (direction, distance) at a single frame in a video clip, while the distribution G is an extracted Gaussian component that represent an object or spatial visual concept. Using this distance measure we can decide which of the

[2]Note that the robot does not know the words *colour*, *shape* and *distance* specifically, the features are named $\textit{feature}_1$, $\textit{feature}_2$, etc. The words are only used for the ease of explanation to the reader.

concepts (e.g. $shape_1$, $distance_1$, etc.) holds as a predicate for which objects in the video clip by finding the minimum distances between observations and concepts in each feature space. For example, in a video clip where there exists an object with id $= i$, and its shape value is closest to the visual concept $shape_j$. Then, we say that the visual concept $shape_j$ holds as a predicate for object i in that particular frame. This is achieved by adding the predicate $shape_j(i)$ to the list of predicates at that frame, the same applies for the other object properties and spatial relations. An example from the LUCAS robot dataset is shown in Figure 3.8 where the generated predicates for each object property and relation are added to the figure for that particular frame. This process is repeated for every frame in a video clip to encode the visual world and represent each object and relation at every frame in the video clip. The next step in learning robot-action concepts is to represent changes in object properties and spatial relations using spatio-temporal graphs. Note that the changes will always relate to *locations*, *directions* and *distances* in the datasets used in this thesis. However, the same approach can be applied to changes in other features such as *shapes* when assembling parts to create an object with a new shape, or *colours* when painting an object with a different colour.

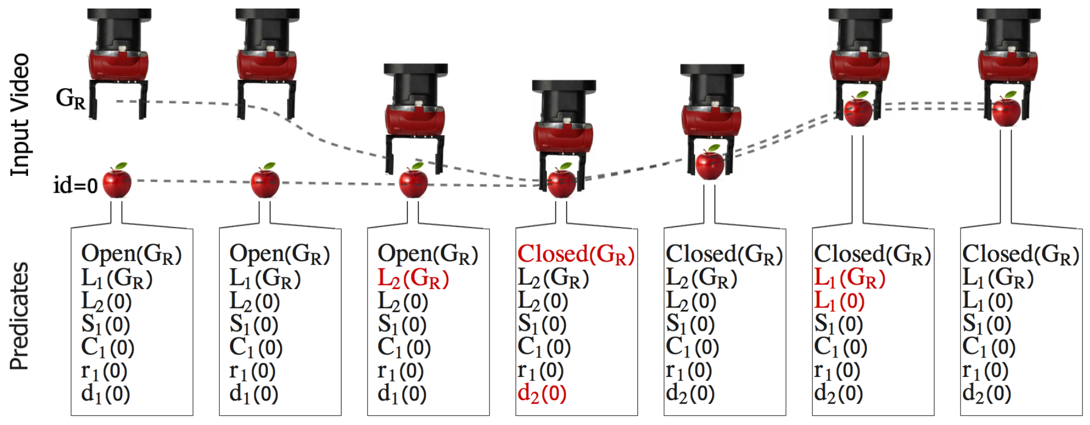

Input Video
Predicates
G_R
id=0
Open(G_R) L_1(G_R) L_2(0) S_1(0) C_1(0) r_1(0) d_1(0)
Open(G_R) L_1(G_R) L_2(0) S_1(0) C_1(0) r_1(0) d_1(0)
Open(G_R) L_2(G_R) L_2(0) S_1(0) C_1(0) r_1(0) d_1(0)
Closed(G_R) L_2(G_R) L_2(0) S_1(0) C_1(0) r_1(0) d_2(0)
Closed(G_R) L_2(G_R) L_2(0) S_1(0) C_1(0) r_1(0) d_2(0)
Closed(G_R) L_1(G_R) L_1(0) S_1(0) C_1(0) r_1(0) d_2(0)
Closed(G_R) L_1(G_R) L_1(0) S_1(0) C_1(0) r_1(0) d_2(0)

Figure 3.8: Encoding the visual world into a list of predicates. An example showing LUCAS picking up an apple. Seven frames are shown along with their encoded predicates printed in the boxes below. In this video, there exist two location concepts (L_1=initial gripper location, and L_2=initial apple location), one shape concept (S_1=apple shape), one colour concept (C_1=red), one direction concept (r_1=above), and two distance concepts (d_1=far, and d_2=touch), along with the two predefined gripper states (*Open*, and *Closed*). The change in a predicate value is indicated with red in the boxes. The change in value means that the object property or relation has now a smaller distance with a different concept in this frame. Using these predicates the robot now has an internal representation of the visual world and how it changes at every frame.

Spatio-temporal graphs

Spatio-temporal graphs are Directed Acyclic Graphs or DAGs (1975) comprising of three layers. These graphs were used in the literature to model human activities as presented by Sridhar *et al.* (2010a; 2010b), Gatsoulis *et al.* (2016a) and Duckworth *et al.* (2016b); an example of these graphs is shown in Figure 3.9. The three layers of the spatio-temporal graphs are: (1) the object layer: used to represent the objects in the scene with a single node per object, (2) the spatial layer: used to represent the Qualitative Spatial Representations (QSRs) between pairs of objects with a single node per spatial relation, and (3) the temporal layer: used to represent changes in spatial relations using Allen's Interval Algebra (1983) which comprises of thirteen basic relations between time intervals that are distinct, exhaustive, and qualitative.

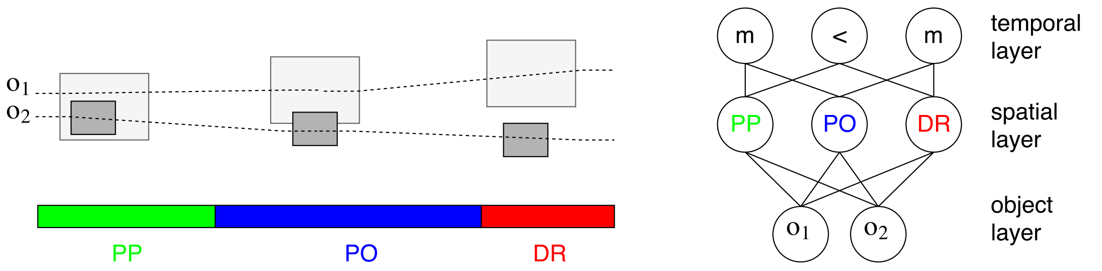


Figure 3.9: Spatio-temporal DAGs representation. (top-left): two object o_1 and o_2 moving away from each other with every frame. (bottom-left): the QSRs between the two objects at every frame using Region Connection Calculus 5 by Cohn and Gotts (1996) with relations Proper-Part (PP), Partially-Overlapping (PO) and Distinct-Regions (DR). Three intervals (PP, PO, DR) are generated for this video, where in each interval a different QSR holds between o_1 and o_2. (right): the spatio temporal DAG for the two moving objects, the temporal relations between the QSRs using Allen's Interval Algebra (1983) the relations are meets (m) and before (<).

We extend the spatio-temporal graphs in two ways: First, more object properties and spatial relations are encoded into each layer of the graph structure to allow for richer and more complex representation of the scene. Second, our DAG uses extracted object and spatial concepts that the robot learns by clustering the video clips as opposed to predefined ones such as Region Connection Calculus 5 (RCC5) by Cohn and Gotts (1996) in the example shown in Figure 3.9. In our spatio-temporal DAGs, we abstract the temporal changes in the world into states, where each state represents a constant qualitative configuration of the visual world. This final step helps the robot in abstracting the changes in the environment into a sequence of states that can be executed once observed, which enables the robot to repeat/learn the observed actions. An example demonstrating our graph representation is shown in Figure 3.10. In this example, we extend the one shown in Figure 3.9 by allowing object properties to change in the video, e.g. the colour property of object o_1 is constantly changing from white to grey through the video. Also, we use learned distance concepts to replace predefined RCC5 relations.

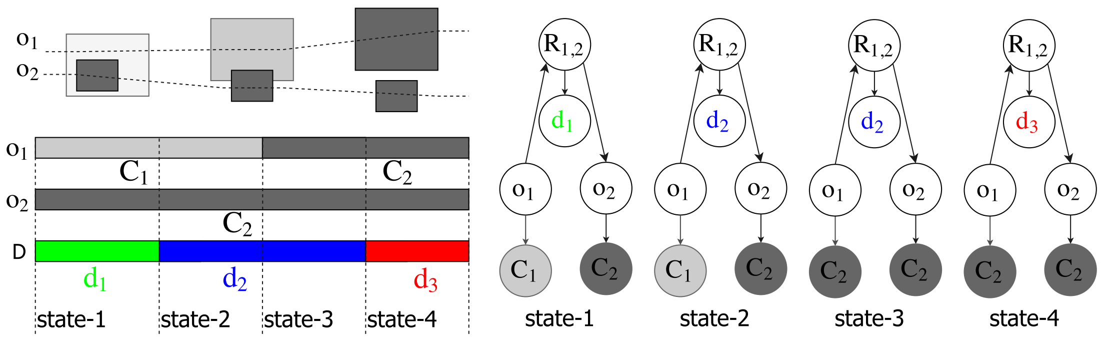


Figure 3.10: Extended DAG representation for two moving objects with changing properties: (top-left) two objects o_1 and o_2 moving away from each other, and object o_1 is changing its colour from white to grey. (bottom-left) this video has two colour concepts (C_1=white, and C_2=grey), and three distance concepts (d_1= touch, d_2=near, and d_3=far). The three rows show the values of object colours (o_1, o_2) and distance (D) at every frame, forming a number of intervals. By splitting the intervals vertically whenever a change occurs in any of them, we generate a sequence of states. Four states are generated for this video clip, where each state represents a constant qualitative configuration of the visual world. (Right) the extended DAG representation for this video clip showing the four states.

In our extended DAG representation shown in Figure 3.10: at any given time in a video clip, the state of the visual world is represented as a directed acyclic graph (DAG) with two layers: object and relational layers, while the temporal layer is discarded and replaced with the use of a sequence of states. The object layer is used to represent the objects in the scene, where each object is encoded with one node along with object property nodes connected to it, e.g. *colour*, *shape*, *location*, etc. The relational layer is used to represent the spatial relations between pairs of objects, where the relations are encoded with a relational node $R_{i,j}$ between objects i and j, along with relational feature nodes connected to it, e.g. *directions*, *distances*, etc. In Figure 3.10, the visual world is represented with a single object property (*colours*) and a single relation (*distances*). This video contains two colour clusters (C_1=white and C_2=grey) and three distance clusters (d_1=touch, d_2=near, and d_3=far). The value of each property node at each frame is decided using the generated predicates presented in the previous section §visual world encoding, which is the internal symbol of a visual concept. For example, in the first frame of the video clip in Figure 3.10, the object o_1 is represented with the node o_1 along with a *colour* node with a value of C_1. Similarly, object o_2 is represented with node o_2 and a *colour* node with value C_2, while the relation between objects o_1 and o_2 is represented with the node $R_{1,2}$ and a *distance* node with value d_1. By omitting consecutive repetitions of identical DAGs, i.e. consecutive frames that have the same states of the world, we obtain a sequence of unique DAGs that represents the video clip. In other words, a new DAG is used to represent the state of the world whenever a node changes label from one visual concept to another. We will refer to each of these DAGs as states, since they describe constant qualitative configurations of the visible objects. The video clip in Figure 3.10 contains four states, each of which has a unique DAG that describes a different constant configuration of the visual world.

An example of our graph representation for a simple "pick up" action performed by LUCAS (the Baxter robot) is shown in Figure 3.11. The object node corresponding to the *gripper* is distinguished and is special as it is assumed to form a particular pre-known object type that has only the *location* and the *state* feature nodes connected to it. The *location* feature node of the gripper is similar to the object one, i.e. assigned a location concept, while the *state* node can be either *Open* or *Closed* to indicate the state of the gripper. The top part of Figure 3.11 shows an input video clip where the Gripper (G_R) approaches the Apple (id=0) and picks it up. In this video, all visual features remains constant except for the location and distance features as was previously shown in Figure 3.8. Note that these concepts were not given to the robot, but learned as described previously in §Object related concepts. The gripper and the apple tracks in the top part of the figure are shown with different colours (blue and red) based on which location concept the track is closest to. The distance between each track and location concept is measured using the Mahalanobis distance and the value of the track is assigned with the closest concept (L_1 or L_2). The second part of Figure 3.11 shows a time bar (interval representation) for each object and relation in the scene. The third part (States) shows how a new state is created for every change in a visual concept. A total of four states are created to represent this input video clip, i.e. four states are needed to represent the "pick up" action if the robot arm is positioned directly above the object. Using this extended DAG the robot now has an internal representation to model and learn the different actions. Note that the training data for these actions is collected by asking volunteers to drive the robot arm to perform a given task while a camera records the changes in the environment. The next step is to mine this graph structure into sub-graphs to represent the different objects, relations, and actions.

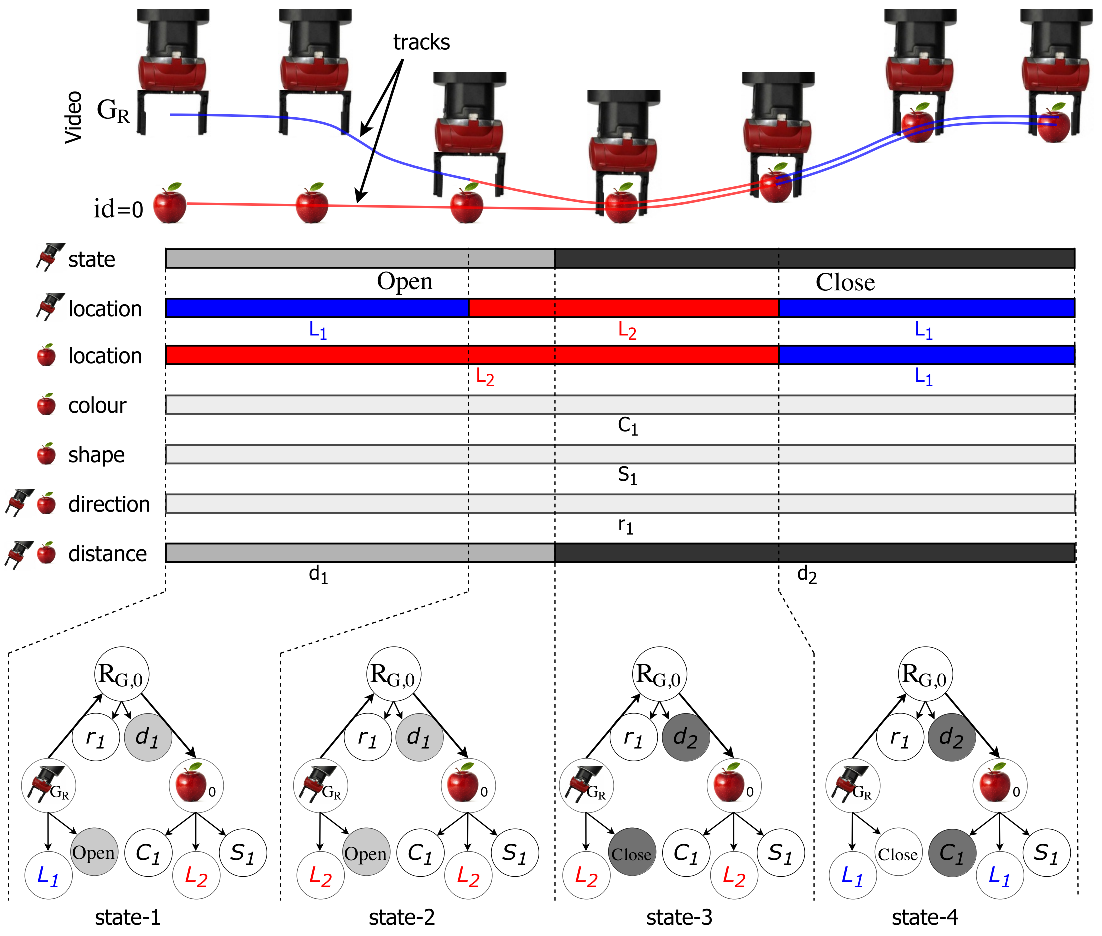


Figure 3.11: Extended DAG representation for a “pick up” action performed by LUCAS comprising of four states. In the first state (state-1) the Gripper is *Open*, and both Gripper and Apple are at different locations and are far from each other, hence their location nodes in the extended DAG representation are assigned different concepts (L_1 and L_2), and their distance node is assigned with the far distance concept d_1. In state-2, as the Gripper approaches the Apple, it gets close enough to the Apple that its location node changes label from L_1 to L_2. In state-3, the distance relation between the two objects gets changed to d_2=touch and the Gripper closes its fingers and change its state to *Closed*. In the final state (state-4) as the gripper lifts the apple up, the position of both objects become closer to the initial location of the gripper (L_1), hence both location nodes switch labels from L_2 to L_1. Note that the nodes are coloured with grey and contain the gripper and apple images for expository purposes only.

Extracting concepts from spatio-temporal DAGs (graphlets)

The principle we use for learning the mapping between language and vision, i.e. bootstrapping the robot's knowledge in both natural language and perception, is to seek frequent co-occurrences of words and sub-graphs extracted from the spatio-temporal DAG of each video clip. The idea is to relate words to fragments of the visual representation of the world. Ideally we would like to perform the learning on all possible sub-graph structures, but this remains an ambition for the future. In this thesis, we steer the learning towards (1) *object* properties, by extracting all connected sub-graphs involving objects nodes and their properties, (2) *spatial relations* between pairs of objects, by extracting all connected sub-graphs from relational nodes R and their properties, and (3) *robot actions*, by extracting sequences of sub-graphs that contain the gripper, the moving object, and the relational nodes that connects the gripper node with this object node. We will refer to these sub-graphs as *graphlets*, where each graphlet has at least one *connection node* (denoted with c) that is used to connect graphlets together. Connecting different graphlets together enables the robot to reconstruct a complete spatio-temporal DAG. This ability will be used later to enable i) the execution of commands, ii) learning of words' meanings, and iii) learning grammar rules. The generated graphlets from the "pick up" example are shown in Figures 3.12, 3.13, and 3.14. The extracted *robot action* graphlets are used as templates for the different actions that the robot observes. These action graphlets can be joined with other object and relational graphlets to assemble a full DAG that the robot can understand and execute. For example, the pick up action graphlet learned from the previous example can be joined with any combination of object property graphlets to pick up that object.

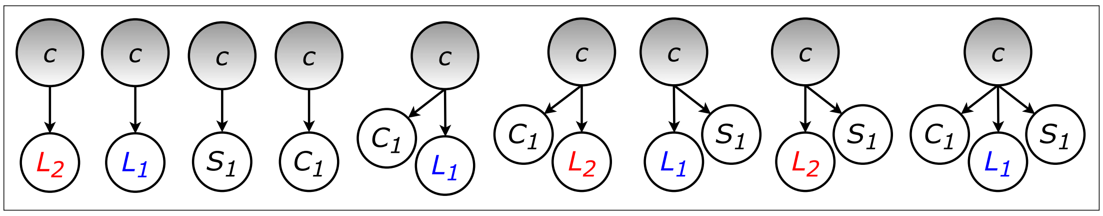


Figure 3.12: All object property graphlets extracted from the DAG in Figure 3.11. From left to right the graphlets represent the initial location of the apple (L_2), initial location of the gripper (L_1), apple shape (S_1), apple colour (C_1), and combinations of these concepts. Note that the connection nodes (c) are highlighted with grey for clarification purposes only.

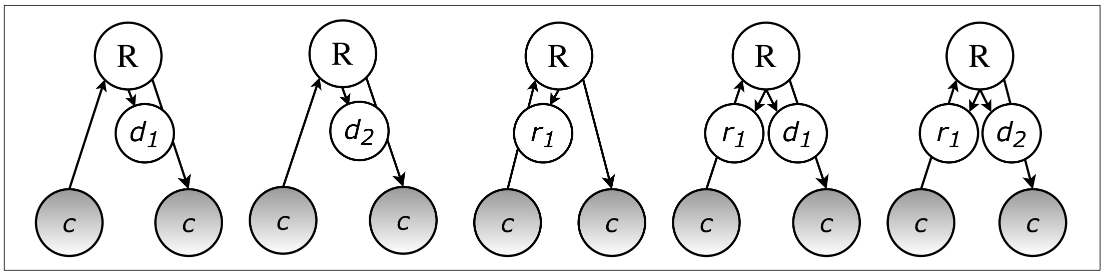


Figure 3.13: All relational graphlets extracted from the DAG in Figure 3.11. From left to right the graphlets represent the far distance (d_1), touch distance (d_2), above direction (r_1), and combinations of these concepts. Note that the connection nodes (c) are highlighted with grey.

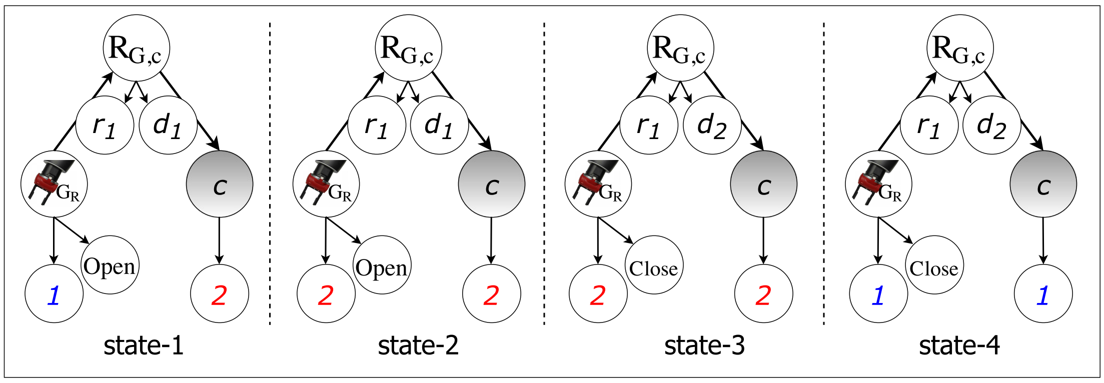


Figure 3.14: The action graphlet extracted from the pick up example in Figure 3.11. Note that only one action graphlet is extracted from each DAG, i.e. the four states are part of a single action graphlet. In this action graphlet, the gripper starts at state-1 with an *Open* state, a different location concept than the object, also above (r_1) and far away (d_1) from the object. At state-2 the gripper changes its location to be the same as the object. At state-3 the gripper touches the object (d_2) and closes its gripper. Finally, at state-4 the gripper carries the object to its initial location. Note that the property nodes 1 and 2 are used to signify two different location concepts, but any two locations can be integrated here by connecting a location graphlets. The connection nodes (c) are highlighted with grey.

3.3.5 Human activities

To learn temporally-extended human activities, the pose of humans within the environment is detected and tracked (as explained in section 3.2.1) along with the positions of the learned objects of interest (as shown in section 3.2.2). Then, the observations are encoded into a number of qualitative spatio-temporal abstractions as presented by Duckworth *et al.* (2017). This encoding condenses noisy observations of arbitrary spatial positions into semantic low-level qualitative descriptors. This allows the system to compare observations based upon key qualitative features and learn common patterns in an abstracted space, instead of their metric positions which can arbitrarily differ. For example, in a "making coffee" activity, the exact spatial position of a person reaching for a mug is not as useful for learning the activity as a qualitative representation of a hand approaching a mug.

We introduce the Qualitative Spatio-temporal Representations (QSRs) used by Duckworth *et al.* (2017) to encode detected pose-object sequences, and computed by a publicly available ROS library by Gatsoulis *et al.* (2016a; 2016b) which I co-authored. A QSR is an abstraction from exact quantitative observations in a particular feature space into qualitative states that hold between the human's pose and objects in the environment. The three representations used are: First, Ternary Point Configuration Calculus (TPCC) by Moratz and Ragni (2008) which qualitatively describes the spatial arrangement of a point relative to two others. That is, it describes the *referent*'s position relative to the plane created by connecting the *relatum* and *origin*. Relations in TPCC are triples of ⟨ { front, back }, { left, right, straight }, { distant, close } ⟩. Second, Qualitative Trajectory Calculus (QTC) by Delafontaine *et al.* (2011) which represents the relative motion of two points with respect to the reference line connecting them, and is computed over consecutive timepoints. For two objects o_1, o_2, it defines the following three relations (objects are abstracted to their centroids when computing QSRs). : $\{o_1$ is moving towards o_2 (symbol $-$), o_1 is moving away from o_2 $(+)$, o_1 is neither moving towards nor away from o_2 $(0)\}$. Third, Qualitative Distance Calculus (QDC) by Clementini *et al.* (1997) which expresses the qualitative Euclidean distance between two points based on defined distance thresholds. A set of QDC relations localises a person with respect to reference landmarks, while changes in the relations can help explain relative motion. An illustration of the three QSRs relative to two objects is shown in Figure 3.15. Note that these QSRs are given to the robot to enable it to learn about human activities, unlike the ones learned in robot-actions.

Once each human pose-objects sequence is converted into a set of qualitative relations (one per frame), we perform a temporal abstraction using Allen's Interval Algebra (1983). This compresses repeated qualitative relations at adjacent frames into an *interval* representation, maintaining the relation and duration information. Secondly, temporal relations are computed between temporally connected intervals to create an *Interval Graph* as presented in Duckworth *et al.*(2017), where nodes represent intervals (relations holding between a set of objects) and

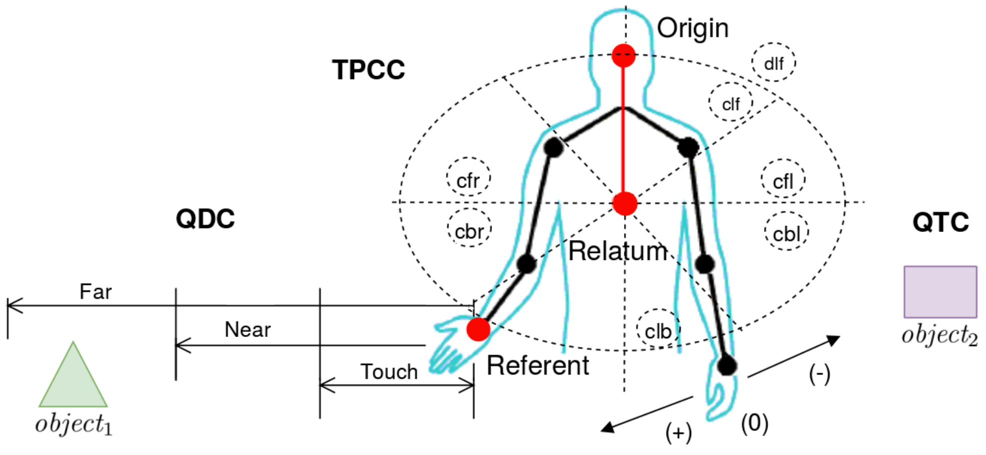


Figure 3.15: QSRs representations; (**bottom left**) QDC between right hand and $object_1$. (**centre**) Subset of the TPCC system between right hand and torso-head plane. (**right**) QTC (relative motion) between left hand and $object_2$. Figure taken from Alomari *et al.* (2017a).

directed arcs link nodes with the temporal relation. Given a corpus of Interval Graphs, one per human detection, a set of unique k-length paths are extracted from the graphs as features for some small k (usually ≤ 4), where each feature represents a small set of temporally-connected spatial relations between a person and some object (likewise ≤ 4). This unique set of features is considered as a discrete *vocabulary*, and thus bag-of-words descriptors of activities (called *activity feature vectors*) can be computed for each detection. This bag-of-words representation is different from the traditional bag-of-words used normally in document analysis in that it maintains some temporal information within its structure.

Latent Dirichlet Allocation (LDA) presented by Blei *et al.* (2003) is used to discover the unique activities (*topics*) observed by LUCIE the mobile robot as presented in Duckworth *et al.* (2017) which I co-authored. This model has proved successful in problems with large corpora not exclusive to document analysis. A topic, a probability distribution over the vocabulary of features, is a conceptual model of a human activity, and thus it is considered as a candidate visual concept representing the different human activities. It should be noted that the Human activity analysis work in this Chapter has been performed as a collaborative work within the STRANDS project consortium (2016). Therefore, the activity modelling itself should not be considered a contribution of this thesis. For more details please refer to the work presented by Duckworth *et al.*(2016a; 2017).

3.4 Continual Learning of Visual Concepts

In our incremental learning process, the robot is introduced to new visual concepts over time, e.g. new faces, spatial relations, human-activities, etc. In this section, we describe the unsupervised incremental techniques used to update the learning of both simple and complex visual concepts when the robot is provided with new observations, i.e. a new short video clip that the robot recorded of its environment.

3.4.1 Simple visual concepts

As discussed in the Concept extraction section 3.3, the term *simple concepts* is used in this thesis to describe visual concepts that can be detected in single observations. For example, human related concepts in 3.3.1, object related concepts in 3.3.2 and finally spatial related concepts in 3.3.3. To incrementally create and update these visual concepts we use an Incremental Gaussian Mixture Model (IGMM) technique presented by Song and Wang (2005). The IGMM is used to create newly observed concepts and update previously learned ones, which is not a straightforward process because visual concepts can vary across different video clips. This variation can happen due to a number of reasons such as different lighting conditions when the videos were recorded, observing the same concept from different view points, occlusions, etc. For example, the colour *red* may be represented with two different Gaussian models in two videos due to differences in lighting conditions between the videos. Another example can be that the face of a person might look different due to observing that person from a different angle, hence modelled with two different Gaussian components. To address this issue, an Incremental Gaussian Mixture Model (IGMM) approach is used to merge or create newly observed visual concepts. The IGMM technique works in two steps: (i) decide whether an observed visual concept has been seen before using two statistical tests, the W-statistic and the Hotelling's T^2 tests, and (ii) update the visual concept if it has been seen before, otherwise create a new one, i.e. create a new Gaussian component in that feature space which is equivalent to learning a new visual concept in the feature space.

Incremental Gaussian Mixture Model (IGMM) works by processing a single video clip at a time. It is also applied to a single feature space (colours, shapes, etc.) at a time, i.e. the visual concepts in each feature space are updated and created independently from the visual concepts in other feature spaces. The IGMM algorithm and implementation details are presented in Appendix A. In this section we will describe how to formulate the learning of simple visual concepts into an IGMM problem. The IGMM takes as inputs the old and the new Gaussian components, the old being the visual concepts that were previously learned from all processed video clips so far, and the new ones being the ones generated from the newly observed video clip. An example of learned (old) visual concepts in the colour feature space are shown in Figure 3.16 (a). This feature space happened to have three learned visual concepts (*red, green, blue*) with *frequencies* (N_1, N_2, N_3). The frequencies are part of the IGMM process and are used to keep track of how often each component was updated (more details on the use of these frequencies are provided in Appendix A). When the IGMM receives the newly observed visual concepts from a new video clip as shown in Figure 3.16 (b), it measures the similarities between the old concepts in (a) and the new concepts in (b) looking for the best match between them as presented in Figure 3.16 (c). Two significance tests are used to measure the similarities between the new and old Gaussian components: the *W statistics* test: which tests the similarity of the covariance matrices; and the *Hotelling's* T^2 test: which tests the similarity between the mean vectors of the two Gaussian components. The output of the IGMM technique is shown in Figure 3.16 (d), every updated candidate concept gets its *frequency* updated as shown in the *red* and *green* Gaussian components, which means that the robot has seen these visual concepts before, learned them, and now updated them when they are observed again. The concepts that did not get updated in the process have their *frequencies* remain constant as in the Gaussian component *blue*, which means the system did not observe this concept in this new video. The new concepts in (b) that did not match with any old concept in (a) will be added as new visual concepts in that feature space with a *frequency* equal to 1, as in the Gaussian component *yellow*. By applying the IGMM process on every new observation, the robot is able to connect new visual concepts with previously learned ones even if they appear slightly different across multiple video clips, as in the colour *red* and the face of a person examples described earlier.

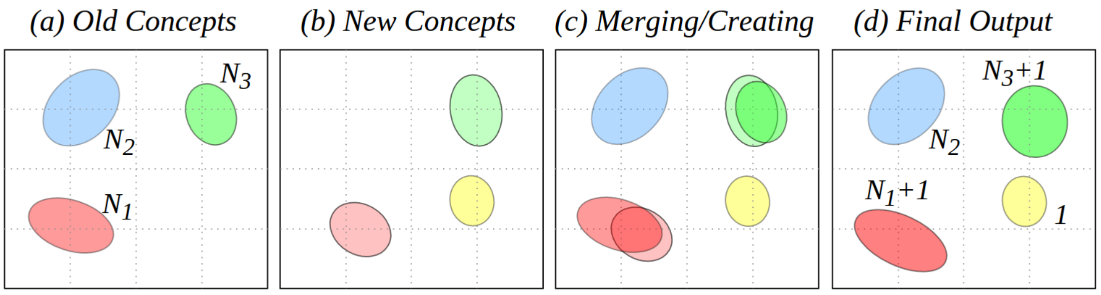


Figure 3.16: IGMM on the simplified HSL colour space (Note that for simplicity HSL feature space is abstracted and shown as a 2D space and not in 3D).

3.4.2 Complex visual concepts

The term *complex visual concepts* refers to concepts that manifest over longer sequences of observations, i.e. can not be learned or recognised using a single frame in a video clip. For example, the robot action concepts presented in section 3.3.4, and the human activity concepts presented in section 3.3.5. For these complex concepts more sophisticated techniques are applied to update learned concepts and create new ones when necessary.

The robot action concepts are represented with action graphlets as described in section 3.3.4, where each action graphlet consists of a sequence of states (spatio temporal DAGs) that each contains the gripper object node and one other object node with a property that changed its label from one visual concept to another (see Figures 3.11 and 3.14). The incremental learning of these graphlets is enabled by the use of graph matching techniques. The incremental learning of graphlets starts by processing one observation at a time, i.e. each video clip is processed separately in four steps to extract the action graphlets. First, extract and update the visual concepts using the IGMM technique. Second, encode each frame in the new video clip into a list of predicates as described in §Visual world encoding. Third, Generate the spatio-temporal DAGs by omitting consecutive repetitions of identical DAGs to obtain the states of the world as described in §Spatio-temporal graphs. Forth, extract the action graphlet from the generated spatio-temporal DAG for the new video as described in §Extracting graphlets. Once the new graphlets are obtained from the new video, each of them is compared against the previously observed/learned ones looking for a match; if no match is found the graphlet is added to the list of all learned graphlets so far. The comparison between newly observed and previously learned graphlets is performed using graph matching techniques implemented in a Python library named NetworkX by Schult and Swart (2008). The matching is achieved by comparing edges and nodes in each graphlet searching for an exact match.

For human activity concepts, the generative LDA model is incrementally updated using Variational Bayes Inference by Hoffman *et al.* (2010). For new observations, the process is divided into two folds: (*i*) the multinomial distribution representing the observed activity over the current set of topics is computed, then (*ii*) the topic distributions over the vocabulary are updated using this new observation. New code words can be added to the vocabulary if they do not already exist, and the topic distributions are uniformly initiated. This allows the robot to efficiently update its model of activity concepts using a single pass over the data, optimising both storage and computation complexity making it ideal for incremental lifelong learning situation. It should be noted that the Human activity analysis work in this Chapter has been performed as a collaborative work within the STRANDS project consortium (2016). Therefore, the incremental activity modelling itself should not be considered a contribution of this thesis. For more details please refer to the work presented by our group in Alomari *et al.* (2017a).

3.5 Discussion

In this section, we discuss the main contributions presented in this chapter in the field of incremental learning of visual concept for robotic systems. We also discuss the limitations and assumptions made in this chapter to enable the learning of both simple and complex visual concepts.

3.5.1 Main contributions

Below is a list of the main ideas and contributions that were presented in this chapter:

1. The use of Gaussian components along with the Bayesian Information Criterion to learn and represent simple visual concepts (colours, faces, shapes, locations, directions and distances) allows for efficient learning from noisy real-world data.

2. The incremental learning of simple visual concepts by employing the Incremental Gaussian Mixture Models (IGMM) technique enables our robot to learn from large data in an incremental and memory efficient manner, which is a step closer towards achieving a human-like life-long learning of simple visual concepts for robots.
3. The extension of spatio-temporal directed acyclic graphs (DAG) representation is also a key contribution of this work. The extension includes adding objects and relational properties to object nodes, allowing for more complex representation of dynamic scenes.
4. The learning framework of visual concepts is easily transferable to learn from different robots, environments and sensing modalities. Also, the learning framework is robust/suitable to learn from noisy real-world data, with partial observations from varying view points of objects and people.

3.5.2 Assumptions: Simple visual concepts

In order to learn about the simple visual concepts, this work assumes that the visual feature spaces are defined beforehand. We define seven continuous visual feature spaces: faces, garment colours, object shapes, object colours, object locations, relative directions and relative distances. The robot starts with the knowledge of how to compute and extract these feature values from an input video clip, but with no pre-given knowledge regarding the number of unique concepts in each feature space or the words used to describe them in natural language, or any prior discretisation of any of the feature spaces. The selection of these features was influenced by the data collected from the robots' environments, which included different objects, people, spatial relations and actions. However, more features can be added without the need to modify the system architecture of our learning framework, as the clustering and learning of simple visual concepts in each feature space is done independently from the remaining feature spaces using the GMM, BIC and IGMM techniques.

3.5.3 Assumptions: Complex visual concepts

The learning of complex visual concepts, i.e. robot actions and human activities, relies on a number of assumptions. For robot actions, the robot is assumed to be equipped with the knowledge needed to build spatio-temporal DAGs from video sequences, and how to mine these DAGs to generate graphlets which represent the different robot actions. Similarly, for learning concepts in human activities, the robot is assumed to be capable of recognising and tracking people poses in the scene using the OpenNI tracker (2016) and the Convolutional Pose Machine (CPM) technique. It is also assumed that our robot can encode raw measurements of people poses and objects into three different QSRs; the Ternary Point Configuration Calculus (TPCC), the Qualitative Trajectory Calculus (QTC) and the Qualitative Distance Calculus (QDC). The computation of these QSRs is performed using a publicly available library. These QSRs are used as features/vocabulary to model the human activities using the LDA technique. Note that the selection of these QSRs was based on domain knowledge of the types of human activities expected in an office environment where LUCIE the mobile robot was deployed, more QSRs can be encoded to model human activities in different environments without the need to modify the learning framework of human activity concepts.

It is also worth noting the reasons behind using both spatio-temporal DAGs and LDA to model complex visual concepts. The spatio-temporal DAGs were chosen for modelling the robot actions because of their ability to represent an action as a sequence of events that the robot can repeat. This enables the robot to model and execute actions by simply observing them, which is one of the aims of this work to teach robots about actions by demonstrating them, yet it is limited to learning relatively simple actions that are repeated in the same order such as picking up objects and moving them around. On the other hand, for temporally-extended human activities, a more elaborate encoding and more sophisticated clustering mechanism (such as LDA) are needed to capture the variations in human activities. Humans tend to perform the same action in various different ways, for example "making coffee", people add the ingredients in different orders and the given label of all of these actions is making coffee. LDA assumes exchangeability between codewords (or grahlets) when modelling an observation. This implies that the specific encoding of an action's codewords is subject to permutation and variation. This allows for modelling similar actions, performed in a different order, into the same topic (or a single visual concept), but at the same time, prohibiting the ability to execute/repeat the action as the specific ordering of how to perform the action is lost.

3.5.4 Scalability

The use of Incremental Gaussian Mixture Model (IGMM) and Variational Bayes techniques allows our robots to learn from large data incrementally, whilst efficiently updating its model of the different visual concepts. Both IGMM and Variational Bayes Inference work by processing one observation (a single video clip or a batch of clips) at a time, without the need to look at them again to update existing visual concepts or learn new ones in the future, optimising both storage and computational complexity.

3.5.5 Loosely-supervised learning

Even though the techniques used to extract the visual concepts from their feature spaces are unsupervised, e.g. GMM, BIC, IGMM, LDA and Variational Bayes, the learning architecture of visual concepts in this thesis is named loosely-supervised as opposed to unsupervised. The reason of naming it this way is that the visual feature spaces (e.g. faces, shapes, etc.) were provided to the robot. We believe a fully unsupervised system should be able to generate new feature spaces when needed, allowing the robot to learn much more complex visual concepts without the need to provide the feature spaces to represent them. But for now it remains as an ambition for future work.

3.5.6 Superlatives, comparatives, collectives and arity relations

The learning of visual concepts is limited to concepts that can be directly represented in the predefined visual feature spaces. Therefore, the learning of concepts that require logical operations and more complex representations is not included in this thesis. This includes, for example superlatives as in "rightmost", "highest", "largest", etc., comparatives as in "further", "higher", "larger", etc. collectives of objects as in "tower", "stack", "column", etc., and arity relations as in "between". The learning of such concepts remains open for future work by building on our existing methodology for learning visual concepts.

3.5.7 Simulated robot

In the simulated robot environment/dataset shown in Figure 3.17, the state of the world is assumed to be fully observable. The extracted feature values are reformed to mimic real world data by adding noise to observations. The added noise to each feature value was generated from Gaussian distributions with zero mean and appropriate covariances in every dimension.

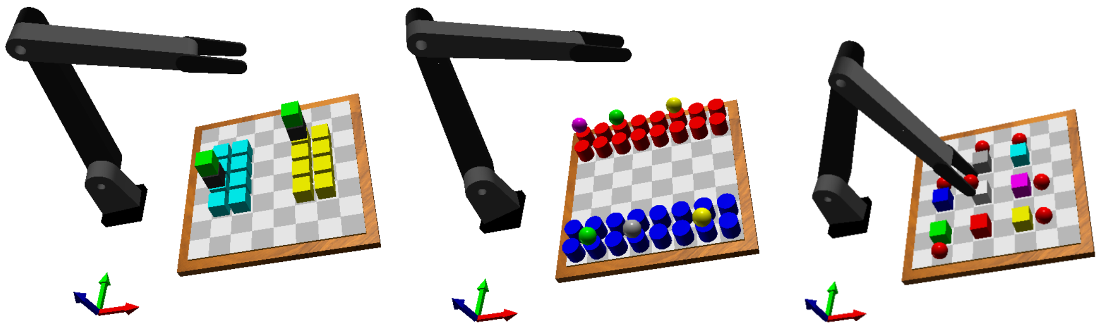


Figure 3.17: Examples from the simulated robot environment (Extended *Train-Robots* dataset).

3.5.8 Mobile robot

For its basic operations, the mobile robot is equipped with a base-mounted laser scanner that is used to model the physical environment as a 2D occupancy grid where occupied cells indicate static objects, allowing localisation, mapping and navigation, as shown in Figure 3.18. For this purpose, an off-the-shelf ROS-packages developed by the STRANDS European project consortium (2016) is used.

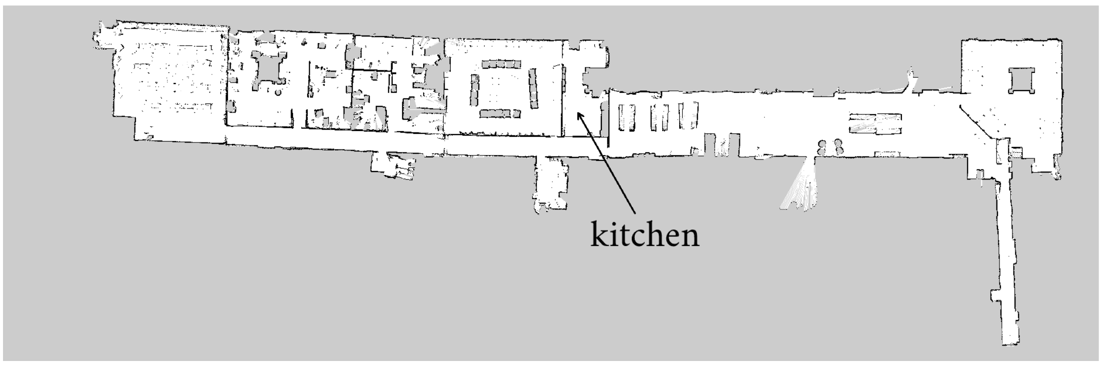


Figure 3.18: The generated map of level 9 in the School of Computing, University of Leeds.

Chapter 4

Natural Language Grounding

Understanding how children learn the meaning of words has long fascinated cognitive and linguistic scientists. This problem is equally important in the field of AI to enable robots to learn the meaning of words they encounter. In the field of psychology, *referential uncertainty* is often thought to be the most important aspect of word learning. As described by Quine (1960), referential uncertainty is a general problem everybody faces when trying to learn a new language. Quine poses this problem as a scenario where an anthropologist studies an isolated tribe. When members of the tribe see a rabbit, they shout "gavagai". Referential uncertainty refers to the problem the anthropologist encounters when hearing the word "gavagai" for the first time and has no idea what the word means. In principle the meaning of the word "gavagai" could be anything from a visual concept of a physical object or entity, features of the surrounding environment, social and historical facts, etc. The space of possible meanings for this word is essentially infinite and the question is how can the anthropologist learn the meaning of this new word. In this work, the robot is assumed to be in a similar position to the anthropologist. The robot has to learn the meaning of words without being able to ask a direct question as to what each individual word means, as the robot is assumed not to know the language initially, and therefore it has to learn the meanings from observations. To enable the learning, we limit the space of possible meanings to concrete observable concepts, i.e. concepts that can be measured here and now when the word in mentioned. More precisely, these are concepts that have the following three properties: i) are related to physical entities and relations between these entities as opposed to abstract concepts like social facts; ii) exist in the scene when the word is mentioned; and, iii) can be measured with the available sensors on the robot. This simplified version of referential uncertainty is usually referred to in the robotics community and NLP literature as the "natural language grounding" problem.

Natural language grounding is a term used to refer to the problem of learning the meaning of words in other domains by mapping words to concepts in these domains, e.g. grounding words to vision-related concepts as presented by Siskind (1996), Roy (1999), Steels (2001), Spranger (2015), and Dukes (2014), or grounding words to robot actions as presented by Lauria (2002), Matuszek (2013), Misra (2015), Chai (2014), Tellex (2011), Chen (2011), and Siskind (2015). In this thesis, the grounding problem is tackled across several domains or feature spaces simultaneously. These features include the human, object, spatial and robot related concepts that are learned incrementally using unsupervised techniques as described in Chapter 3 and as presented by our work in Alomari *et al.* (2016a; 2016b; 2017a; 2017b; 2017c).

4.1 Grounding framework

In this section, we describe how the robot performs grounding of words in natural language sentences to the learned visual concepts in order to communicate effectively with humans in its environment. First, it is essential that the robot gets a natural language description of what it is learning about to perform the language grounding. Ideally we would like our robot to have a speech recognition modality and the capacity to learn about people, objects, qualities and actions, but this remains an ambition for the future. At present, we collect multiple natural language textual descriptions of video snippets recorded by the robot. The descriptions are provided by volunteers and online crowd-sourcing tools such as Amazon Mechanical Turk. Examples of the collected natural language descriptions are shown in Figure 4.1 from each of the recorded datasets used in this thesis.

The aim is to learn the meaning of key words in the sentences by mapping/grounding them to visual concepts that represent their meaning. For example, we aim to learn that the grounding of the word "Eris" shown in Figure 4.1 (top left) is the visual concept in the face

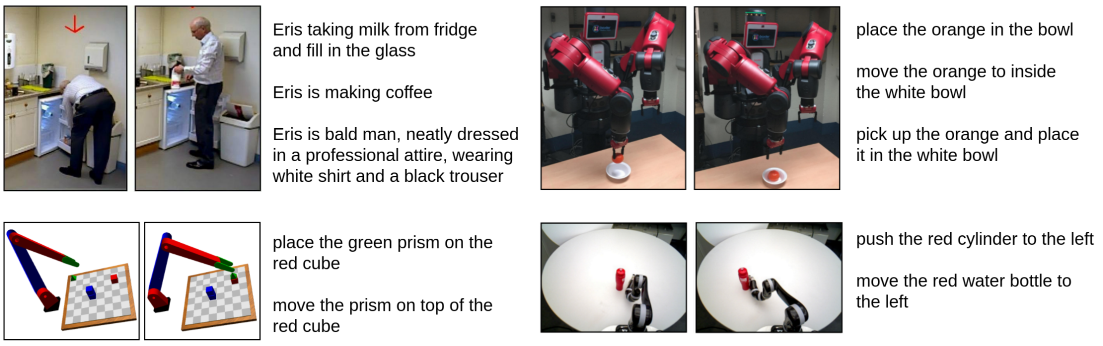

Eris taking milk from fridge and fill in the glass
Eris is making coffee
Eris is bald man, neatly dressed in a professional attire, wearing white shirt and a black trouser
place the orange in the bowl
move the orange to inside the white bowl
pick up the orange and place it in the white bowl
place the green prism on the red cube
move the prism on top of the red cube
push the red cylinder to the left
move the red water bottle to the left

Figure 4.1: Examples of natural language descriptions collected for four different datasets.

feature space that represents how Eris looks like. Similarly, we aim to learn that the grounding of the word “black” is the visual concept representing the black colour in the HSL feature space. The language grounding process, i.e. learning the mapping from language to vision, involves several steps that are all shown in Figure 4.2. The process starts by first abstracting both raw linguistic and visual inputs into a number of concepts as shown in the *pre-processing* step in Figure 4.2. Then, we aim to find the correct mappings between linguistic and visual concepts by following the four steps in the *Grounding Language to Vision* block in Figure 4.2: 1) building a *language-vision* correlation matrix that measures the probability of associating linguistic and visual concepts together in a way that is inspired by human language development; 2) generating hypotheses that map concepts from language to concepts in vision; 3) filtering the generated hypotheses using case analysis and a number of predefined rules to reduce their number by ruling out some incorrect ones; and, at the end 4) validating these hypotheses through mental simulations and graph matching techniques to find the correct grounding from language to vision. These steps are described in detail in the following sections.

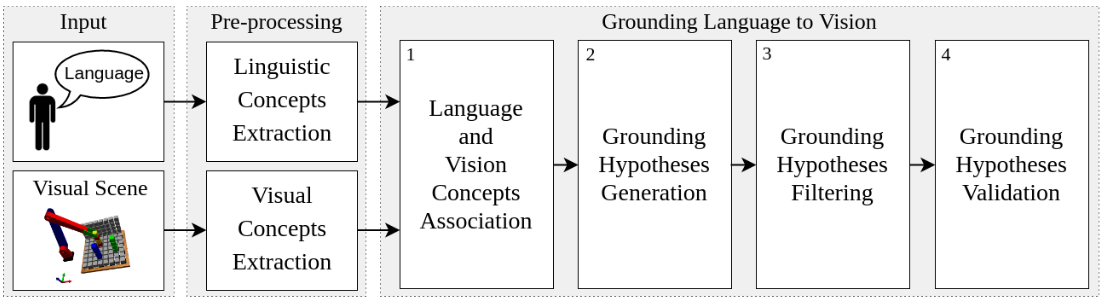

Input
Language
Visual Scene
Pre-processing
Linguistic Concepts Extraction
Visual Concepts Extraction
Grounding Language to Vision
1
Language and Vision Concepts Association
2
Grounding Hypotheses Generation
3
Grounding Hypotheses Filtering
4
Grounding Hypotheses Validation

Figure 4.2: The language grounding learning framework from sentences and video clips.

4.2 Language and Vision Concept Extraction

The learning of language grounding is performed on a single pair of visual and linguistic inputs at a time, i.e. a video clip and a sentence describing it. For each video and sentence pair shown in Figure 4.1, we aim to match words from the input sentence to their visual representations in the input video clip. To achieve this goal, we need to first represent both domains, language and vision, in a way that enables the mapping between them.

For the vision domain, the input video clip is processed to generate a set of visual concepts as described in Chapter 3. The newly generated visual concepts from this input video are merged with the previously learned ones. The mapping between new and old visual concepts is achieved using the IGMM and graph matching techniques described in *Continual Learning of Visual Concepts* in 3.4. The set of all learned visual concepts from all feature spaces are accumulated together into a single list $\mathcal{V} = \{v_1, \ldots, v_u\}$, where v_i is a visual concept, e.g. a colour, a relation, a robot action, etc., and u is the total number of visual concepts across all video clips. The list $\mathcal{V}$ is updated with every new video clip. I.e. the list $\mathcal{V}$ holds the cumulative knowledge the robot has gained about the visual world and is updated incrementally.

For the linguistic domain, each sentence is processed independently from other sentences even if they are describing the same video. The process starts by converting the input text to all lower case and removing any punctuation (as this is not explicitly present in spoken language). We then extract all possible n-grams from a sentence with $n \leq M$, where M is the longest sequence of n words to form an n-gram. An n-gram is a sequence of n consecutive words in a single sentence, and is considered to be our representation for language, i.e. a linguistic concept. The use of n-grams allows the learning of multi-word descriptions such as '*pick up*', '*light blue*' and '*bottom left corner*'. The list of all unique n-grams across all input sentences are combined into a list $\mathcal{N} = \{\delta_1, \ldots, \delta_b\}$, where δ_i is an n-gram, and b is the total number of unique n-grams in all input sentences. The list $\mathcal{N}$ is updated with every new input sentence.

The robot now has an intermediate representation for both vision and language domains. This representation transforms knowledge from continuous spaces to bounded discrete ones, the list $\mathcal{V}$ represents the cumulative knowledge of all visual concepts, and the list $\mathcal{N}$ for all linguistic concepts. These two lists are used to learn the associations between language and vision.

4.3 Language and Vision Concept Association

According to recent studies in the field of child language development by Owens Jr (2016), every child has a bank of words in their brain. For every child, the bank is of different size. Some children need to hear a word up to seven hundred times before they understand its meaning and are able to use it correctly. Other children, especially ones with learning difficulties, need lots more, and therefore repetition is key in teaching children about the meaning of a word. In this thesis, we rely on repetitions between words and visual concepts to teach our robots the meaning of words in the vision domain. The grounding of language is also inspired by an idea from Hebbian theory, which can be summarized as: "*Cells that fire together, wire together*" by Schatz (1992). This idea is translated to "*concepts in language and concepts in vision that appear together, wire together*". As an example, the word '*apple*' and the *apple* shape concept will appear repeatedly and consistently together throughout the input videos and text; therefore the two concepts should be wired together (grounded), while the word '*the*' is not solely consistent with any visual concept and therefore it should not be connected to any.

To measure the consistency of repetitions between concepts in language and vision, we follow the frequentist approach presented by Everitt and Skrondal (2002). We keep track of the number of times an n-gram and a visual concept appear individually, and the number of times the two appear together, across all observed instances. Given the set of all learned vision concepts $\mathcal{V}$ of length u, and the set of all observed unique n-grams $\mathcal{N}$ of length b, we define a *concepts correlation* matrix $\mathbf{K}$ of size $b \times u$ and with n-grams as rows and visual concepts as columns as shown in Figure 4.3.

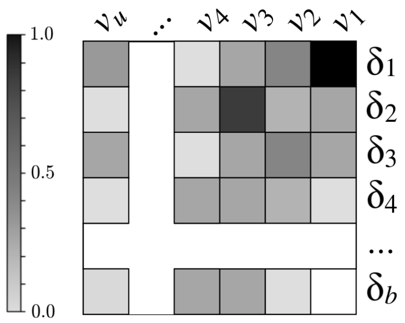


Figure 4.3: The *concepts correlation* matrix $\mathbf{K}$ of size $b \times u$, where each observed n-grams is a row, and each visual concept is a column in the matrix.

The values in the *concepts correlation* matrix $\mathbf{K}$ are computed using Equation 4.1 that contains two parts: the maximum of two frequentist terms representing strength of association between an n-gram (δ) and a vision concept (v), and an exponential function representing the learning curve, where $\lambda(.)$ is a count function, and τ is the decay rate constant.

$$\mathbf{K}(\delta, v) = \underbrace{\max\left(\frac{\lambda(\delta, v)}{\lambda(\delta)}, \frac{\lambda(\delta, v)}{\lambda(v)}\right)}_{\text{strength of association}} \underbrace{\left(1 - e^{-\frac{\min(\lambda(\delta),\lambda(v))}{\tau}}\right)}_{\text{learning curve}} \tag{4.1}$$

The first part of Equation 4.1 computes the strength of associating an n-gram (δ) to a vision concept (v) by counting the number of times a vision concept and a language concept are observed together, normalised by either the number of times the vision concept is observed or the language concept is observed, i.e. the strength of associating δ to v or v to δ, where $\lambda(.)$ is the count function, i.e. (i) $\lambda(v)$ counts the number of times a vision concept v is observed in all processed videos, (ii) $\lambda(\delta)$ counts the number of times an n-gram δ is observed in all processed text, while (iii) $\lambda(v, \delta)$ counts the appearance of both concepts v and δ together in all processed input video-sentence pairs, i.e. both v and δ has to appear in a video clip and its description to increment this counter. The value of this part of the equation ranges between 0 and 1. It is equal to 1 if both concepts v and δ are constantly appearing together, and is equal to 0 if they are never seen together in the same pair of inputs. We use both terms $\left(\frac{\lambda(v,\delta)}{\lambda(v)}, \frac{\lambda(v,\delta)}{\lambda(\delta)}\right)$ to concentrate on the less-observed of the two concepts, improving the quality of the multi-to-multi association and preserving the richness of language. I.e. by using the maximum of both terms, we allow the robot to learn multiple words describing the same vision concept and learn multiple vision concepts representing the same word. For example, if the word "Eris" appeared in the description of 20 video clips, 10 of which person-A was featured in and the other 10 person-B was there, and also both person-A and B never appeared in any other videos. I.e. $\lambda(\delta) = 20$, $\lambda(v_A) = 10$, and $\lambda(v_B) = 10$. For both person-A and B, the number of times where both the word "Eris" and the vision concept appear together is equal to 10, i.e. $\lambda(\delta, v_A) = \lambda(\delta, v_B) = 10$. It is clear that by normalising on the less-observed of the two concepts (language or vision) generates a strength of associating "Eris" with a value of 1 with person-A, i.e. $\max\left(\frac{\lambda(\delta,v_A)}{\lambda(\delta)}, \frac{\lambda(\delta,v_A)}{\lambda(v_A)}\right) = \max\left(\frac{10}{20}, \frac{10}{10}\right) = 1$. The same applies for person B.

The second part of Equation 4.1 (the exponential component) represents the certainty or the learning curve of concepts. The aim of using this component is to penalise the proposed strength of associating concepts that have been observed only a few times. Consider the scenario where the robot observes an *apple* for the first time, every word in the input sentence is equally likely to be describing this new shape, and the probability of associating this visual concept with all the words in the input sentence is equal to 1 (i.e. the first part of Equation 4.1 is equal to 1). The same applies for linguistic concepts observed for the first time. Hence, we need a mechanism to penalise the probability for concepts that were observed only a few times. The second component in Equation 4.1 is an exponentially decaying function towards a limiting value. Such functions have the form $\left(y = y_\infty + \alpha e^{-\frac{t}{\tau}}\right)$, where $t = \min\left(\lambda(v), \lambda(\delta)\right)$, y_∞ is the limiting value of the function when $t \to \infty$[1], α is a constant that helps setting the function value when $t = 0$, and τ is the decaying rate constant. The higher the value of the constant τ, the slower the system converges to the y_∞ value, and visa versa as shown in Figure 4.4. In Equation 4.1, the limiting value (y_∞) is set to 1, and the constant α is set to -1, creating an exponential function that ranges from 0 at ($t = 0$) and 1 at ($t \to \infty$). The decaying rate constant value (τ) is chosen to be in the range of 5 to 10; a concept has to be observed 25 to 50 times in order for the second part of Equation 4.1 to equal 1. A detailed sensitivity analysis on the selection of these parameters is presented in the Experimental Results chapter.

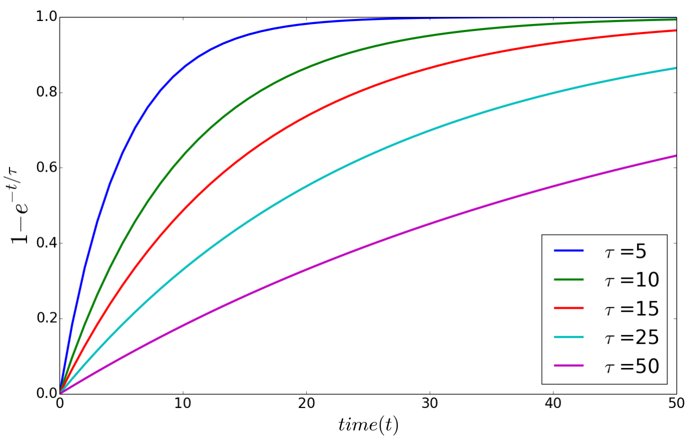


Figure 4.4: The effect of changing the value of the decay constant τ on the convergence to y_∞.

[1]The ∞ value in the exponentially decaying functions is usually assumed to be five times the value of τ, i.e. at ($t = 5\tau$). At this value, the exponential component is too small ($e^{-5} = 0.0067$) and can be ignored, leaving the exponential decaying function value equal to the limiting value ($y = y_\infty$).

Once the new input video-sentence pair is processed, the elements of the *concepts correlation* matrix $\mathbf{K}$ are updated according to Equation 4.1. Each element in the matrix $\mathbf{K}(i,j)$ holds information of the strength of associating the i^{th} n-gram to the j^{th} visual concept. This information is used in the next section to generate the initial hypotheses that map/ground natural language to vision.

4.4 Grounding Hypotheses Generation

Grounding language to vision is a challenging task for those who do not speak the language. Going back to the anthropologist example, the meaning of the word "gavagi" is not necessarily limited to a single visual concept. It is possible that "gavagi" means different things in the more general case, which means that the mapping between words and vision concepts is not always one-to-one. For example, certain words can refer to different visual concepts like names, e.g. *Eris* can be the name of any person regardless of their look. Also, some visual concepts can be described with different words, e.g. a block shape can be referred to in the English language with *block, brick, slab, bar,* etc. To learn the grounding of language to vision, we search for the highest correlations between n-grams and visual concepts that feature in each video clip and description, allowing multi-to-multi associations to preserve the richness of natural language. The associations are measured using the *concepts correlation* matrix $\mathbf{K}$ as described in the previous section. Defining a target function $\mathcal{A}$ which has $\mathcal{A}(\delta, v) = 1$ if the association (δ, v) is selected as a grounding candidate and 0 otherwise, we can formulate the problem of multi-to-multi language-to-vision grounding as solving an integer program with the objective function:

$$\max_{\mathcal{A}} \sum_{\mathcal{N}\times\mathcal{V}} \mathcal{A}(\delta, v)\, \mathbf{K}(\delta, v). \tag{4.2}$$

We maximise the objective function with the following constraints:

– $\sum_{\mathcal{N}\times\mathcal{V}} \mathcal{A}(\delta, v)/(b * u) < \epsilon$, keeping sparsity of the groundings by forcing the number of selected groundings to be below some small ϵ (set between 5 and 10%) of the total number of possible groundings. A detailed sensitivity analysis on ϵ is performed in the Experimental Results chapter.

– $\sum_{\mathcal{N}} \mathcal{A}(\delta, v) \geq 1$, $\forall v \in \mathcal{V}$, forcing the assignment of at least a single n-gram to each of the learned visual concepts. This helps ensuring that each visual concept gets at least one word to describe it even in noisy data. The reason why we do not enforce the same rule on n-grams is because some words can relate to no vision concepts in the scene, such as *function words*, e.g. articles, pronouns, pro-sentences, auxiliary verbs, etc.

Solving this integer program results in assigning a number of highly-correlated n-grams to each visual concept. An example for solving the integer program for matrix $\mathbf{K}$ is shown in Figure 4.5, where $\mathcal{A}(i, j) = 1$ (black) for every chosen grounding, and $\mathcal{A}(i, j) = 0$ otherwise. The error in this process gets rectified through filtering, validation and continual learning processes which will be discussed in the following sections.

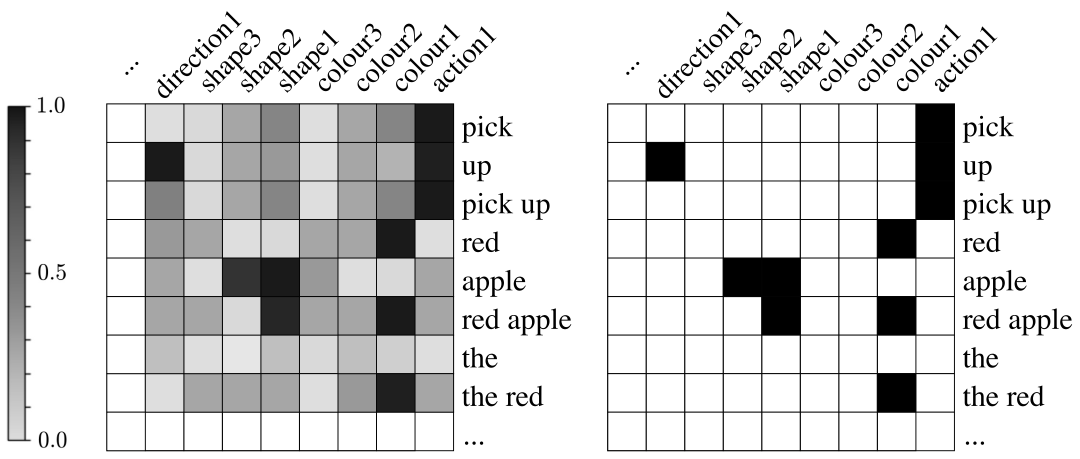


Figure 4.5: Grounding hypotheses generation. (left) The *concepts correlation* matrix $\mathbf{K}$, where each observed n-gram is a row, and each visual concept is a column. The value of each element varies from 0 to 1, where 0 means the two concepts were never observed together. (right) The target function $\mathcal{A}$ that results from solving the integer program for the matrix $\mathbf{K}$. Integer programming allows for multi-to-multi associations between n-grams and vision concepts. For example, ‘up’, ‘apple’ and ‘red apple’ each is mapped to two visual concepts.

4.5 Grounding Hypotheses Filtering

By using n-grams as linguistic concepts, we end up with a number of n-grams that map to the same visual concept, some of which are incorrect. For example, the n-grams (‘*red*’, ‘*the red*’, and ‘*the red apple*’) will all be connected to the same *red* colour visual concept with high probability. Therefore, we need to filter out the incorrect ones '*the red*' and '*the red apple*' from the target function $\mathcal{A}$ and keep only the correct groundings between *red* colour and the word '*red*'. This is achieved by case analysis. Consider the case of whether to accept the assignments in $\mathcal{A}$ of n-gram ab, consisting of smaller n-grams a and b (e.g. the 2-gram '*the red*' consists of the 1-grams '*the*' and '*red*'). Let v_{ab}, v_a, v_b be the visual concepts assigned to the n-grams δ_{ab}, δ_a and δ_b respectively. There are four possible cases shown by the rules below (4.3 to 4.6) from which we can figure out which ones of these n-grams are incorrect. The accepted assignment hypotheses are shown on the right side of the arrow. For example, in Equation 4.3 all three n-grams are assigned to the same visual concept ($v_{ab} = v_a = v_b$), then we accept the hypothesis for the bigger n-gram $\mathcal{A}(\delta_{ab}, v_{ab}) = 1$ and filter out the smaller n-grams $\mathcal{A}(\delta_a, v_a) = \mathcal{A}(\delta_b, v_b) = 0$.

$$v_{ab} = v_a = v_b \quad \rightarrow \quad \mathcal{A}(\delta_{ab}, v_{ab}) \tag{4.3}$$

$$v_{ab} = v_a \neq v_b \quad \rightarrow \quad \mathcal{A}(\delta_a, v_a), \mathcal{A}(\delta_b, v_b) \tag{4.4}$$

$$v_{ab} = v_b \neq v_a \quad \rightarrow \quad \mathcal{A}(\delta_a, v_a), \mathcal{A}(\delta_b, v_b) \tag{4.5}$$

$$v_{ab} \neq v_a \neq v_b \quad \rightarrow \quad \mathcal{A}(\delta_{ab}, v_{ab}), \mathcal{A}(\delta_a, v_a), \mathcal{A}(\delta_b, v_b) \tag{4.6}$$

Rule (4.3) filters out the smaller incorrect n-grams, by allowing complex n-grams to subsume their constituent ones when all of them are equal. The intuition behind it can be seen in examples like the n-grams '*pick up*', '*pick*' and '*up*' where we want to keep the longer n-gram '*pick up*' and remove the smaller ones '*pick*' and '*up*' if they are all grounded to the same visual concept. Rules (4.4, 4.5) filter out the larger incorrect n-grams. The intuition behind it is we do not want the robot to use more words than necessary to describe a concept, such as '*the red*' to describe the *red* colour. Rule (4.6) states that if the n-grams are connected to different concepts, keep all of them. This rule can be used to learn phrasal verbs where their meaning is different to their individual components. For example, the phrasal verb '*break down*' is different to both '*break*' and '*down*'. These rules will filter some of the incorrect groundings. Also, they will not stop different synonyms from connecting to the same vision concept. For example, '*cyan*' and '*sky blue*' could share the same vision concept, because '*cyan*' is not a constituent of '*sky blue*'. After filtering out some of the incorrect groundings (example shown in Figure 4.6) the robot ends up with a number of candidate grounding hypotheses that require validation; details of the validation process are presented in the following section.

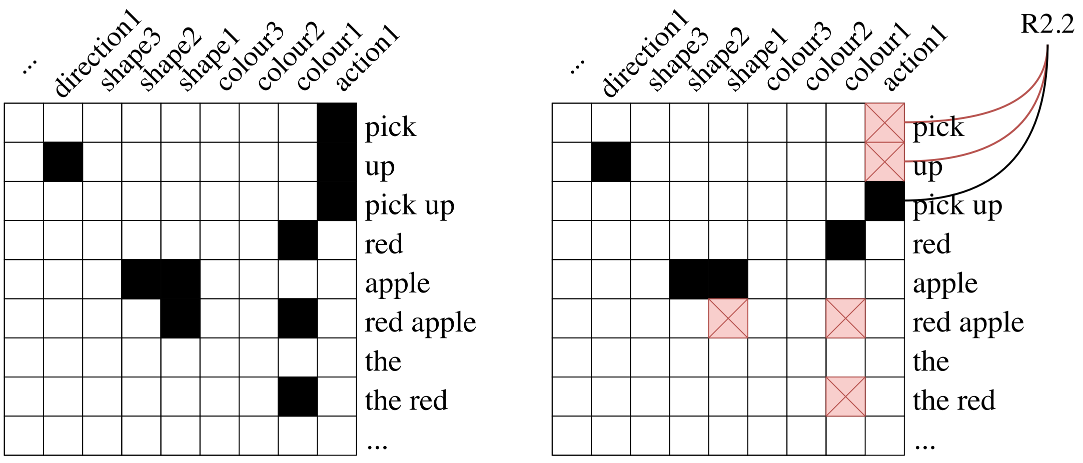


Figure 4.6: Filtering the grounding hypotheses. (left) The target function $\mathcal{A}$ obtained by solving the integer program from the previous example shown in Figure 4.5. (right) The resultant target function $\mathcal{A}$ after filtering the grounding hypotheses using Rules 4.3 to 4.6. The accepted hypotheses where the target function $\mathcal{A}(\delta, v) = 1$ are shown in black, while the rejected hypotheses where $\mathcal{A}(\delta, v) = 0$ are shown in red to highlight them. The grounding hypotheses related to 'pick' and 'up' are rejected based on Rule 4.3, while the hypotheses related to 'red apple' and 'the red' are rejected based on Rules 4.4 and 4.5. In this example, the 1-gram 'apple' is still mapped to two shape concepts after filtering, *shape*1 represents the apple shape, and *shape*2 represents the mug shape. We will show in the following section how we can validate which of these two is the correct grounding.

4.6 Grounding Hypotheses Validation

Once the best grounding hypotheses between language and vision concepts have been selected and filtered, we attempt to validate them by using mental simulations to identify the correct ones. Mental simulation is a term first articulated by Craik (1967) and is used to describe how humans evaluate their environment to better understand the interactions between its components. For example, with a simple glance humans can realise whether a stack of dishes will topple, or whether a branch will support a child's weight. Craik's suggestion is that humans perform quick mental simulations that allow them to imagine different scenarios, and to try out a learned activity in their mind to predict the most likely outcome for each situation. This idea has been used in a computational setting to help robots analyse physics based simulated environments to figure out whether a stack of blocks would topple as presented by Battaglia *et al.* (2013). Our approach is motivated by Craik's proposal, that the brain builds mental models that support inference by mental simulations, to help the robot verify which of the available grounding hypotheses between language and vision are correct and which are not. Imagine the scenario were the 1-gram '*apple*' in the given input sentence "*pick up the apple*" is grounded with two different visual concepts, one representing the shape *apple* in the FPFH feature space, and the other representing something incorrect, e.g. the shape of a *mug* as shown in Figure 4.5. This can occur due to noise or insufficient data, e.g. whenever the robot encounters the word '*apple*' in the input sentence it finds a mug and an apple in the corresponding video clip. The hypotheses validation process developed here aims to find the correct groundings for every n-gram and visual concept if any exist. The validation is accomplished using two steps: First, examining the outcome of adopting each grounding by simulating an environment. Second, comparing the simulated environment with the input video through graph matching techniques.

The mental simulations of the environment is achieved by translating the input text into multiple scenes/graphs as follows: first, substituting all n-grams in the sentence with their grounded visual concepts that were selected and filtered in the target function $\mathcal{A}$ as described in the previous two sections; second, representing each visual concept as a graphlet as shown in Chapter 3 in Figures 3.12, 3.13, and 3.14; third, connecting the graphlets together in different orders to create all possible graph structures (spatio-temporal DAGs). The order in which the graphlets are connected is important and will later map to learning grammar in the following Chapter. We refer to the graphs generated from connecting the graphlets together as *hypothesis graphs*. Each *hypothesis graph* represents a different course of actions taken by the robot in the simulated world that reflects what it thinks the sentence means. For example, if the robot believes the 1-gram '*apple*' in the previous example ("*pick up the apple*") might mean the apple shape, it will pick the apple shape in the simulation. On the other hand, if it assumes it means the mug shape, it will pick up the mug in the simulation. These two hypotheses about the 1-gram '*apple*' will create two different *hypothesis graphs* shown in Figure 4.7.

Each simulated scene (from a sentence) is compared against its corresponding input video sequence. The idea is to look for a match between a simulated scene and the input video. We use the matching between the two as a clue to infer that the hypotheses used to generate the

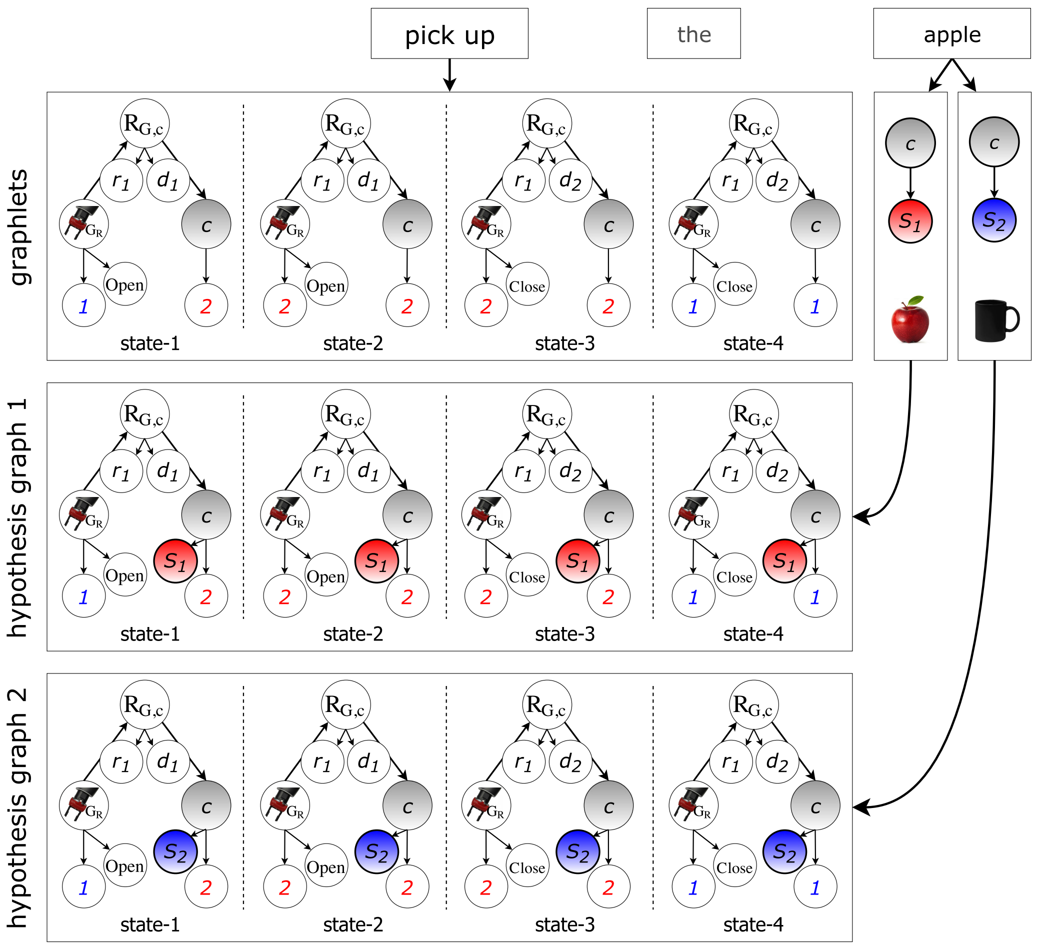


Figure 4.7: Generating *hypothesis graphs* from the sentence "*pick up the apple*". The 2-gram 'pick up' has one candidate vision concept representing the pick up action, 'the' has none, while 'apple' has two (S_1=apple shape, and S_2=mug shape). These vision concepts in their graphlet format are combined to generate two *hypothesis graphs*.

simulated scene are correct. For example, using the hypothesis that relates the word 'apple' to the apple shape visual concept will result in a simulated scene where the robot behaves in a similar way as to what happened in the input video. In other words, the robot picks up the apple object, which is not the case when simulating the mug shape. The matching between this simulated scene and the input video supports the hypothesis that the word 'apple' should be grounded to the apple shape concept (S_1) and not the mug shape concept (S_2), and this is how we validate each of the available grounding hypotheses. The comparison between the mentally simulated scenes and the input video occurs on the graph level. The comparison is achieved by testing if any of the *hypothesis graphs* matches with the spatio-temporal DAG extracted from the input video. The matching is enabled by using an *induced sub-graph* matching technique presented by Howorka (1977). We say that a simulated scene matches the input video if its *hypothesis graph* is an induced subgraph of the spatio-temporal DAG extracted from the input video. For example, we want to validate the grounding hypotheses for the input sentence "*pick up the apple*" with the input video shown in Figure 3.11. The initial grounding hypotheses for this example were generated using the integer program technique shown in Figure 4.5. Then the hypotheses were filtered as shown in Figure 4.6, which reduces the number of potential candidate hypotheses. The robot now has one vision concept grounded to the n-gram 'pick up', none for the 1-gram 'the', and two candidate concepts for the 1-gram 'apple': one concept representing the apple-shape (S_1), and one representing the mug-shape (S_2). To validate these grounding hypotheses, multiple *hypothesis graphs* are generated that reflect all possible combinations. This is done by connecting the *connection nodes* (denoted with c) in both the action graphlet and the object graphlets together as shown in Figure 4.7. The robot then checks which (if any) of the generated *hypothesis graphs* match the input video. Since that *hypothesis graph-1* (shown in Figure 4.7) is an induced sub-graph of the input video DAG (shown in Figure 3.11), then, we say that the hypotheses used to build the *hypothesis graph-1* are validated and will be used to represent the n-grams '*pick up*', '*the*' and '*apple*'. The exact procedure of how to probabilistically accumulate the knowledge of grounding hypotheses is described in the following section. Note that the 1-gram '*the*' is learned to be a *function word* as it has no mapping in the vision domain and the graph was still validated.

4.7 Learning Probabilities of Language Grounding Φ

In this section, we describe how to accumulate the knowledge about all validated hypotheses. For example, if the 1-gram 'apple' was validated 10 times, nine of which were with the shape-apple (S_1), and only once it was validated to be the shape-mug (S_2), then we need a way of saying that these two are not equal. Note that this might happen as our visual concept clustering is unsupervised and does not produce perfect results; also because we learn from noisy data.

Once a *hypothesis graph* is validated as described in the previous section, the robot learns all of the grounding hypotheses used to build it. The accumulation of grounding hypotheses knowledge is performed in a similar way to learning the Part-of-Speech tags of words in Natural Language Processing (NLP) applications. Part-of-Speech (POS) tagging refers to the process of marking up each word in a sentence with an appropriate grammatical category. The grammatical categories vary between languages and can range to hundreds of different types, for example in English a word can be tagged as *noun, verb, article, adjective, preposition, pronoun, adverb, etc.* POS tagging is not an easy task to perform for machines, and is harder to learn than just having a list of words and their POS tags as a training set. This is mainly because some words can be labelled with different POS tags in different sentences, which is quite common in natural languages. For example, the word "*dogs*", which is usually thought of as just a plural noun, can also be marked up as a verb in the sentence "*The sailor dogs the hatch*"[2].

The learning of such POS tags in NLP literature can be divided into two categories, supervised and unsupervised. In the supervised setting, a tagger is trained on a corpus of sentences labelled by a human expert with their equivalent POS tags, like ("*The sailor dogs the hatch*", "*Determiner Noun Verb Determiner Noun*"). The manual labelling of data is a tedious task that hinders learning from large corpora, and is not necessarily available for all languages. In the unsupervised setting, clustering techniques are employed to generate a set of POS tags from unlabelled data by exploiting regularities in natural language and word signatures, i.e. words are clustered based on the similarity of their neighbouring words where each cluster forms a POS tag. While unsupervised POS tagging techniques enable learning from unlabelled data, their performance is usually significantly worse than those of the supervised techniques.

In this thesis, the learning of word tags is performed on unlabelled data and enabled by combining language and vision inputs in a loosely supervised manner. The word tags are assumed to be the visual concepts extracted in Chapter 3 such as *colours, shapes, distances, etc.* Once a grounding hypothesis that connects an n-gram (δ) to a vision concept (v) is validated, the robot updates its knowledge about it. This is achieved by updating the probability of this grounding hypothesis using Equation 4.7, where Φ is the grounding function that maps n-grams to vision concepts ($\Phi : \mathcal{N} \to \mathcal{V}$), P is the conditional probability for a vision concept v given

[2] "dog the hatches" a sailing term that means to lock or make tight the doors and windows.

the n-gram δ, $\lambda(\delta, v)$ counts the number of times the n-gram δ was validated with the vision concept v, and $\lambda(\delta)$ is the total number of times the n-gram δ was validated with any vision concept. For example, if we revisit the example mentioned earlier where the word 'apple' was validated 10 times ($\lambda(\delta) = 10$), 9 of which was with the apple-shape S_1, and 1 of which was with the mug-shape S_2 ($\lambda(\delta, S_1) = 9$, and $\lambda(\delta, S_2) = 1$). Then, the probability of grounding the word '*apple*' to the apple-shape (S_1) is equal to 0.9, i.e. $\Phi(\delta, S_1) = P(S_1|\delta) = 0.9$, and the probability of it being grounded to the mug-shape S_2 is equal to 0.1. The grounding function Φ is important and will later be used in learning grammar rules in the following chapter. Also, it is used to enable our robot to parse and executing commands by mapping language to vision as will be shown in the Experimental Results chapter. The probabilities in the grounding function Φ are updated incrementally for each n-gram as more grounding hypotheses are validated as discussed in the next section.

$$\Phi(\delta, v) = P(v|\delta) = \frac{\lambda(\delta, v)}{\lambda(\delta)} \tag{4.7}$$

4.8 Continual Learning of Language Grounding

For incremental grounding of natural language, the entire pipeline of language grounding is executed again whenever new visual observations and text descriptions are available. This is vital to obtain correct groundings of language to vision as the richness of natural language and the possible noise in the data require continuous re-evaluation of the associations and mapping between language and vision. This incremental process is achieved by the following six steps:

1. Add new rows and columns to the correlation matrix $\mathbf{K}$ to keep track of newly-observed n-grams and newly-learned vision concepts.

2. Update the frequency measure of every observed n-gram and vision concept pair in the correlation matrix $\mathbf{K}$ using Equation 4.1.

3. Re-solve the integer program to generate new grounding hypotheses with the objective function 4.2.

4. Filter the grounding hypotheses using the four developed n-gram case analysis shown in rules 4.3, 4.4, 4.5 and 4.6.

5. Validate the hypotheses using mental simulations by comparing *hypothesis graphs* with the input video spatio-temporal DAG.

6. Update the probabilities in the grounding function Φ for each validated grounding hypothesis using Equation 4.7.

In this grounding framework, there is no need to store the original data, i.e. the input sentences and the raw input video clips. We only need to keep track of the frequencies in the *concepts correlation* matrix $\mathbf{K}$ to perform the incremental grounding of language. In the following section we present an idea that prevents the matrix $\mathbf{K}$ from growing indefinitely.

4.8.1 Habituation of concepts

The *concepts correlation* matrix $\mathbf{K}$ will grow in size as more observations are provided to the robot, more rows for newly observed n-grams, and more columns for newly learned vision concepts. This will result in an increased processing time and memory space requirements to keep track of all language and vision concepts. To address this issue, we built a mechanism inspired by an idea from psychology called habituation. In psychology, the term 'Habituation' refers to the diminishing of an innate response to a frequently repeated stimulus as presented by Bouton (2007). In our case, the stimulus is the observation of a linguistic or a visual concept in an input, and the innate response is the update of the *concepts correlation* matrix $\mathbf{K}$. By applying the habituation concept to our system, the robot loses interest in updating the meaning of a visual or linguistic concept if it observes this concept for many times. I.e. if the robot observes the word 'apple' for more than a certain threshold n_h (i.e. $\lambda(\delta) > n_h$), then, the 'apple' word is assumed to be no longer interesting for learning and its corresponding row is removed from the matrix $\mathbf{K}$. This is clearly a naive interpretation of the habituation idea as we assume a cut off threshold in learning concepts, that once passed the robot loses interest in this concept, when in fact it is a much more complicated process. Also, even though the word 'apple' was removed from the *concepts correlation* matrix $\mathbf{K}$, its grounding knowledge is preserved in the grounding function Φ. It is just that the robot stops being interested in learning what 'apple' means as by now it should have converged to the correct grounding.

4.9 Discussion

In this section, we highlight the main contributions in the field of language grounding presented in this thesis. We also discuss the assumptions made to enable the learning of language grounding, and the learning of function words.

4.9.1 Main contributions

Below is a list of the main ideas and contributions presented in this chapter:

1. Learning the grounding of language to vision from raw textual inputs as opposed to parsed inputs with extracted grammatical categories, like verbs, nouns, adjectives, etc. Our approach enables learning of different languages, even ones that do not have trained parsers for them. Also, we use n-grams as linguistic concepts which enables learning of longer descriptions such as 'light green', 'pick up', 'top left corner', etc.

2. Using extracted/learned vision concepts as candidate groundings of language as opposed to using hard-coded pre-defined concepts.

3. Formulating the language grounding problem into an integer programming one, allowing for multi-to-multi associations between language and vision, preserving the richness of natural language. Also, language grounding is learned in an incremental manner by extending the *concepts correlation* matrix $\mathbf{K}$ with new observations.

4. Using mental simulations and graph matching techniques to validate the grounding hypotheses is the main contribution in this chapter. Also, using habituation idea (inspired from the field of Psychology) to prevent the increase of processing time and memory space requirements for the grounding technique.

4.9.2 Loosely supervised learning of language grounding

Even though the techniques used to learn the grounding of language are unsupervised, such as the integer programming and the mental simulation, the learning architecture of natural language grounding in this thesis is named loosely-supervised as opposed to unsupervised for three reasons. First, the input videos are assumed to include a single action in each, e.g. a single pick up or a single use of the microwave in each video clip. Second, the sentences are assumed to be describing the concrete concepts in the scene, such as the *actions, colours, shapes, etc.* as opposed to abstract concepts. Third, the videos and sentences are temporally aligned beforehand, i.e. the robot knows which sentences belong to which video clips. We believe a fully unsupervised system should be able to learn from long videos and text, i.e. be able to temporally segment the videos and map the segments to sentences automatically. This will allow our system to learn from much more rich sources like *YouTube* videos, but for now it remains an ambition for future work.

4.9.3 Human activities and language grounding

In Chapter 3, we discussed how learning human activities differs from learning robot actions as it requires more elaborate encoding and more sophisticated clustering mechanism to model the variation in each activity class, and how LDA and Variational Base algorithms are used to model these activity classes. Humans tend to perform the same action in various different ways, for example "making coffee", people add the ingredients in different orders and the given label of all of these actions is making coffee. LDA assumes exchangeability between codewords (or grahlets) when modelling an observation. This implies that the specific encoding of an action's codewords is subject to permutation and variation. This allows for modelling similar actions performed in a different order into the same topic (or a single visual concept), but at the same time prohibiting the ability to execute/repeat the action as the specific ordering of how to perform the action is lost. In our grounding framework, the validation process described in §Grounding Hypotheses Validation (4.6) assumes that the robot is capable of executing the action in a simulated world to validate the grounding hypotheses. Therefore, using LDA to learn human activities eliminates the ability of validating the grounding for these visual concepts. Hence, when learning from data containing people performing various tasks we limit the learning framework to the first two steps and use n-grams of length equal to one, $n \leq N, N = 1$ as shown in Figure 4.8. The search for a visual representation that has the capacity to model human activities performed in different orders, and at the same time maintain the ability to repeat/execute the actions is outside the scope of this thesis and remains an ambition for future work in the fields of computer vision and human activity recognition.

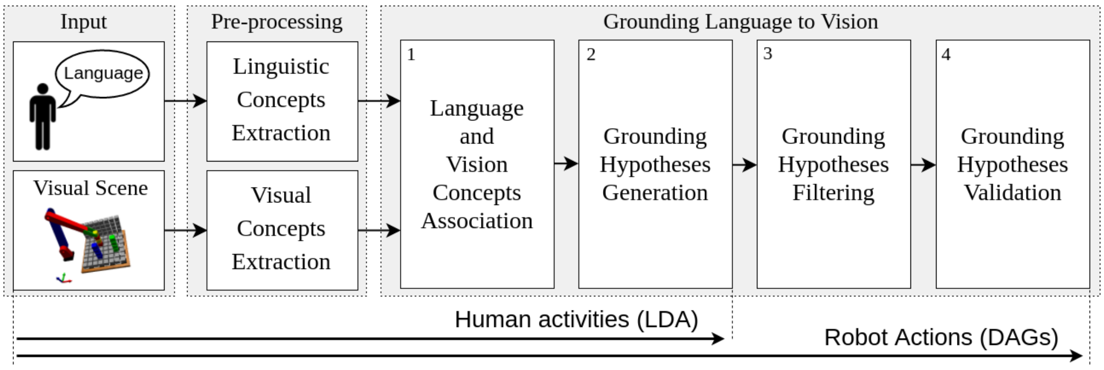

Input
Language
Visual Scene
Pre-processing
Linguistic Concepts Extraction
Visual Concepts Extraction
Grounding Language to Vision
1
Language and Vision Concepts Association
2
Grounding Hypotheses Generation
3
Grounding Hypotheses Filtering
4
Grounding Hypotheses Validation
Human activities (LDA)
Robot Actions (DAGs)

Figure 4.8: Grounding framework for human activities and robot actions.

4.9.4 Function words

To simplify the learning of language grounding in robotics applications, it is common to use a stop word list to remove function words such as 'the' and 'as' from all sentences. But, since we learn from unlabelled data (i.e. avoiding human annotation including stop word lists), we learn such words using the integer programming technique where certain words do not have any mappings with the vision domain such as the word 'the'. This has the same effect as using term frequency-inverse document frequency (tf-idf) weighting to remove function words as presented by Jones (1972).

Chapter 5

Grammar Induction

The human brain is the only precise and complete language processing system currently known to us. Linguists and psychologists have debated for decades on how humans acquire the knowledge of natural language components and how they master speaking and understanding it; and also how much of it is an innate knowledge and how much is learned. However, they all agree that we acquire language in a primarily unsupervised fashion. On the other hand, nearly all computational approaches developed to learn about natural languages are supervised. In particular ones developed to learn the language structure (grammar), relaying on human experts to provide training data labelled with grammar trees. An example is shown in Figure 5.1 for an annotated grammar tree (Robot Control Language tree) from the Dukes (2013) dataset for the sentence "*place the green sphere over the red cube*". These trees are used to train a parser in a supervised manner to model the language structure.

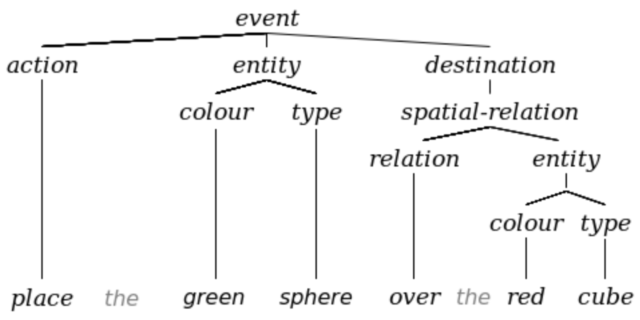


Figure 5.1: Example of an annotated grammar tree used as training data for supervised parsers.

The use of supervised grammar induction techniques was made popular due to the poor performance of unsupervised ones. Unsupervised grammar induction approaches aim to learn the language structure from unlabelled text inputs, making them more desirable to learn from large corpora, and to model languages with no annotated datasets. But, the resultant language model from these unsupervised techniques usually holds little, if no meaning at all, to how the words interact between each other, which is needed by robotic systems to understand and execute natural language commands. An example of a grammar tree generated by an unsupervised system for the previous example "*place the green sphere over the red cube*" is shown in Figure 5.2. The unsupervised system used to generate this tree was presented by Ponvert *et al.* (2011) and was trained on the entire Dukes (2013) dataset. This technique learns a language model via chunking the raw text into smaller parts that shows a repeated pattern throughout the dataset. Using such unsupervised techniques raises two main issues that are hard to fix. First, these methods do not label the chunks in the generated tree; they only output a nested set of brackets defining each chunk of text which carry no meaning to the robotic system. I.e. the robot would not know if a chunk, e.g. "*place the green*" in the Figure below, represents an *entity*, or a *spatial relation*, or an *action, etc.* Second, for systems which do provide labelled brackets, there is the problem of mapping the generated labels with symbols provided by the human expert. This problem is similar to the one faced when evaluating unsupervised clustering techniques, where cluster labels have no inherent link to true class labels and usually do not map one-to-one with the true classes. As can be seen in the example in Figure 5.2, the generated chunks hold no meaning to the robot agent, and are hard (if not impossible) to map back to the sub-tree labels provided by the human expert in Figure 5.1 that are needed by the robot to understand and execute the given command.

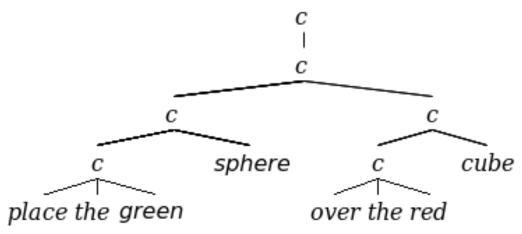


Figure 5.2: Example of an unsupervised grammar tree. The c denotes an unknown label.

5.1 Learning Grammar from Language and Vision

In order to fully understand linguistic commands, the robot needs to learn the grammar rules that govern the sentence structure in natural language. To highlight this, consider the previous example command "*place the green sphere over the red cube*". Even assuming that the robot has a correct visual representation (grounding) for each word in this sentence, the robot still needs an understanding of which object should be placed where. This translates to knowing that the action 'place' changes the location of the 'green sphere' object and not the 'red cube' object, and further, that it needs to change the sphere's position to a final location described by the spatial relation 'over the red cube'. Grammar rules are used to analyse the grammatical structure of a sentence, i.e. establish relationships between head words and words which modify those heads. Providing the system (a robot in our case) with the knowledge needed of how words interact among themselves in a sentences.

In this chapter, we describe our approach for loosely supervised grammar induction from unlabelled inputs. Our approach is developed to parse sentences into grammar trees with meaningful labels and probabilistic grammar rules. The learning of meaningful grammar rules is enabled by the use of both language and vision pairs as inputs, where grammar rules obtain their meanings from the vision domain. The idea of learning grammar rules by mapping them to features in the vision domain has been introduced in the robotics literature before. For example, Dominey and Boucher (2005), Tellex *et al.* (2011), Matuszek *et al.* (2013) and Dukes (2014) have developed supervised systems that can model the structure of natural language commands from vision. A parser is trained to model the language by generating a set of grammar rules that enables the generation of Robot Control Language (RCL) trees from sentences. An example of an RCL tree was shown earlier in Figure 5.1. A more detailed explanation of RCL is provided in the following section. The parsing of sentences into RCL trees enables robotic agents to understand and execute linguistic commands that were not seen before in the training data. In this thesis, the aim is to enable robots to fully understand natural language commands and descriptions without the use of labelled/annotated training data, i.e. without the need of a human expert to label the inputs. In the previous two chapters we showed how to cluster the vision domain to learn a set of visual concepts in Chapter 3, and how to use these visual concepts to learn the groundings of n-grams in Chapter 4. In this chapter, we will use the visual concepts along with the learned language groundings to enable the learning of grammar rules from unlabelled data. In the following sections, the Robot Control Language is described, along with how it is used to enable the learning of grammar rules.

5.2 Robot Control Language (RCL)

Robot Control Language (RCL) is a tree semantic representation for natural language commands. Each sentence is represented as an RCL tree, an example is shown in Figure 5.1, where leaf nodes align to words in the corresponding sentence, and non-leaves are labelled with a predefined set of categories that the robot can understand and execute as presented by Dukes (2014). The RCL elements used in this thesis are presented in Table 5.1. Each element represents one or more of the visual features defined in Chapter 3, which are *object properties* {*colour, shape, location*}, *spatial relations* {*direction, distance*} and *robot actions*. These elements are used to represent the structure of natural language commands for robot manipulation tasks. Although RCL elements used in this work are designed to operate within the context of robot manipulation only, it can be easily extended to other domains such as robot navigation commands as presented by Tellex (2011) and Matuszek *et al.* (2013), or learning from *YouTube* how-to videos as presented by Alayrac *et al.* (2016b), or learning cooking instructions as presented by Beetz *et al.* (2011) and Malmaud *et al.* (2015).

In the robotics literature, the problem of parsing sentences into RCL trees has been formulated as a grammar induction problem. A parser is trained on linguistic commands and their human annotated RCL trees as shown in Figure 5.1. The parser is then used to parse new sentences/commands into trees which the robot can understand and execute. The human annotation of RCL trees is a labour-intensive task that hinders the learning from large datasets, and requires the constant supervision of human experts to provide the trees for learning.

In this thesis, we automatically generate a *vision tree* (Ω) from each input video clip. These vision trees will later substitute the human annotated RCL trees to learn grammar. We define a vision tree Ω as an event tree, i.e. a tree with the $event_v$ element as its head, which consists of three vision elements ($action_v$, $entity_v$, $destination_v$) as shown in Figure 5.3. The v subscript

RCL element	*Description*
event	Specification of a single command. Takes (action, entity, destination) elements as children.
action	Aligned to a verbal group in natural language, e.g. ‘place’.
entity	Specification of a single entity. Takes (colour, shape, location) as children.
destination	A spatial destination. Takes (spatial-relation, location) as children.
spatial-relation	Used to specify a spatial relation between two entities or to describe a location. Takes (direction, distance, entity) elements as children.
colour	Colour attribute of an entity, e.g. ‘red’, ‘green’, ‘light blue’.
shape	Shape attribute of an entity, e.g. ‘pyramid’, ‘apple’, ‘mug’.
location	Location attribute of an entity, e.g. ‘center’, ‘top left corner’.
direction	Direction relation between two entities, e.g. ‘right of’, ‘on top of’.
distance	Distance relation between two entities, e.g. ‘near’, ‘far’.

Table 5.1: The list of all RCL elements used in this thesis. These RCL elements are designed to work in the context of robot manipulation.

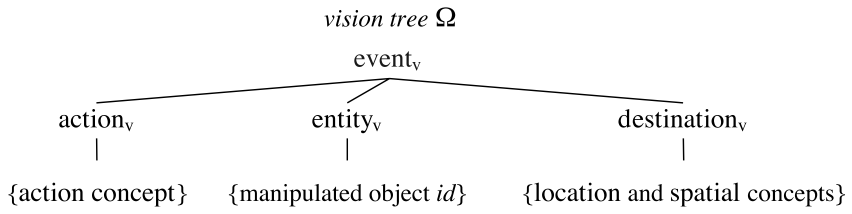


Figure 5.3: Vision tree Ω definition. The vision tree is an event tree, i.e. a tree with the $event_v$ element as its head. The $event_v$ element takes three children $\{action_v,\ entity_v,\ destination_v\}$.

in these elements refers to ‘vision’, to distinguish them from the equivalent RCL elements shown in Table 5.1. The $action_v$ element holds the internal symbol of the action graphlet. Action graphlets are extracted from the spatio-temporal DAG of the input video clip as presented in §Extracting Graphlets (3.3.4). The $entity_v$ element holds the *id* of the object that is manipulated by the robot in the video. The definition and use of object *id*s were presented in §Visual World Encoding (3.3.4). The $destination_v$ element holds the internal symbol of the final location-concept of the manipulated object and the final spatial configuration with other objects in the scene. Learning location and spatial concepts using Gaussian mixture models was presented in §Object related concepts (3.3.2) and §Spatial concepts (3.3.3) respectively. To better explain vision trees, consider the input video example shown in Figure 5.4, which is the video clip paired with the linguistic command "*place the green sphere over the red cube*". We extract a vision tree Ω from this video clip with three elements shown in Figure 5.5. The $action_v$ element holds the internal symbol of the action graphlet extracted from this video clip. This action graphlet was labelled with the internal symbol $action_1$. The $entity_v$ element holds the object *id* of the manipulated object, which is the green sphere with id=0 in this video clip. The $destination_v$ element holds the internal symbol of the location concept of the manipulated object, i.e. the final position of the green sphere, which has the internal symbol $location_4$ in this video clip. Also, it holds the final spatial configuration with other objects in the scene, i.e. the spatial relation[1] with respect to the red cube since it is the only other object in the scene, which is $direction_1(0, 1)$ in this video clip, as indicated on Figure 5.4. In the following section, we show how to use vision trees extracted from input videos to substitute the human annotated RCL trees in learning grammar rules.

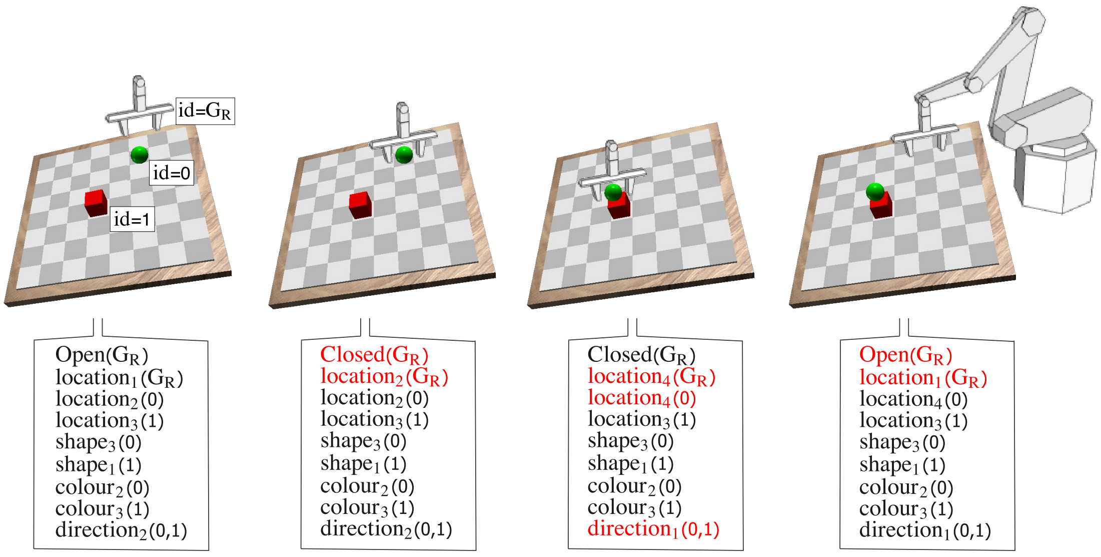

id=G_R
id=0
id=1
Open(G_R) location_1(G_R) location_2(0) location_3(1) shape_3(0) shape_1(1) colour_2(0) colour_3(1) direction_2(0,1)
Closed(G_R) location_2(G_R) location_2(0) location_3(1) shape_3(0) shape_1(1) colour_2(0) colour_3(1) direction_2(0,1)
Closed(G_R) location_4(G_R) location_4(0) location_3(1) shape_3(0) shape_1(1) colour_2(0) colour_3(1) direction_1(0,1)
Open(G_R) location_1(G_R) location_4(0) location_3(1) shape_3(0) shape_1(1) colour_2(0) colour_3(1) direction_1(0,1)

Figure 5.4: The input video clip that we generate for the command "*place the green sphere over the red cube*" encoded with the learned visual concepts shown at different frames.

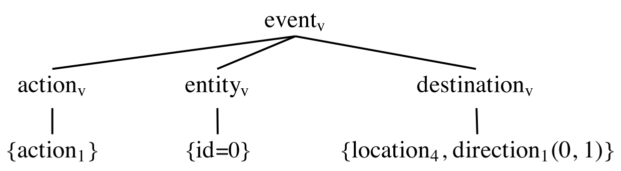


Figure 5.5: The vision tree extracted from the video in Figure 5.4.

[1]Note that in Dukes (2013) dataset the distance feature is not computed, more details in Experiments chapter.

5.3 Generation of RCL trees

To automatically generate RCL trees, we employ the same idea used in validating the language groundings, which is comparing the language model with the vision input. This idea assumes that the input sentences provided to the robot are describing the actions, objects and relations involved in the corresponding input video clip. Therefore, the sentence structure should also reflect/map the features extracted from the input video. We formulate the problem of automatic generation of RCL trees into a search problem as follows. For each input video-sentence pair, we (*i*) extract the vision tree Ω from the input video; (*ii*) generate the set of all possible RCL trees from the input sentence; (*iii*) search for an RCL tree that matches the extracted vision tree Ω. I.e. we aim to find the sentence structure that will result in a match with what happened in the input video. We say an RCL tree matches a vision tree if the values of their corresponding elements are equal, the elements are $\{$*action*-*action*$_v$, *entity*-*entity*$_v$, *destination*-*destination*$_v\}$. Given a match is found between these three elements in a language tree Ψ, we use this language tree to update the robot's knowledge in grammar. The procedure to perform the search for the correct RCL tree Ψ is shown in Algorithm 1, and is divided into four steps (*substitute, group, query*, and *match*). The following sections walk through the entire process using the example "*place the green sphere over the red cube*" shown in Figure 5.5, and shows how the robot obtains a correct RCL tree Ψ from this input video-sentence pair.

Algorithm 1 Automatic generation of RCL trees

```
1: procedure SEARCH FOR CORRECT RCL TREE
2: Variables
3: $\Phi$ is the grounding function $\Phi : \mathcal{N} \rightarrow \mathcal{V}$
4: $\Omega$ is the extracted vision tree from the input video
5: S is the input sentence
6: $\Psi$ is the correct RCL tree
7: Input $\Phi$, $\Omega$, S
8: Output $\Psi$
9: Substitute each word in $S$ with its visual concept using $\Phi : \mathcal{N} \rightarrow \mathcal{V}$
10: Group vision concepts to create RCL elements
11: Query RCL elements with the input video to link the sentence with the video
12: Match RCL elements with the vision tree $\Omega_i$ to find the correct RCL tree
13: if all RCL tree elements pass the matching test with $\Omega$ then
14:     create $\Psi$ from matched RCL tree elements
15: Return $\Psi$
```

5.3.1 Substitute words with visual concepts

For each input sentence S consisting of t words, $S = \langle w_1, \ldots, w_t \rangle$, we substitute each word with the internal symbol of its visual concept using the grounding function Φ learned in Chapter 4. For instance, the sentence $S = \langle \textit{place}, \textit{the}, \textit{green}, \textit{sphere}, \textit{over}, \textit{the}, \textit{red}, \textit{cube} \rangle$, is transformed using the grounding function Φ into $S' = \langle \textit{action}_1, \textit{None}, \textit{colour}_2, \textit{shape}_3, \textit{direction}_1, \textit{None}, \textit{colour}_3, \textit{shape}_1 \rangle$. The grounding function Φ for this example is shown in Figure 5.6 (left). Note that if a word has multiple groundings in Φ, then this process is repeated for all combinations of possible groundings, i.e. a new sentence S' is created for every possible grounding. The word substitution process for this example is shown in the *substitute* section in Figure 5.6 (right).

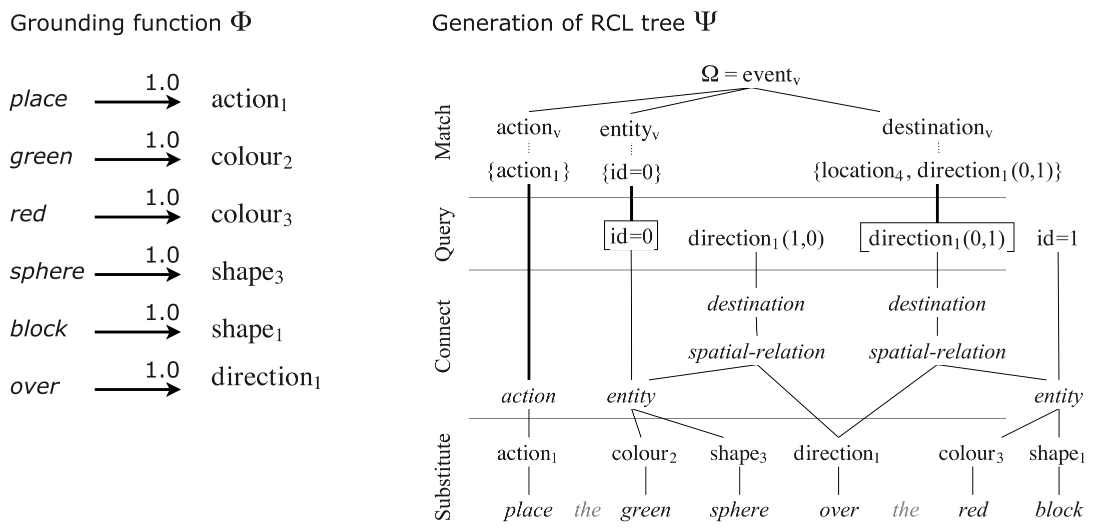


Figure 5.6: Automatic generation of RCL trees. (left) The grounding function Φ showing the probabilities of assigning words to vision concepts. (right) The four steps (Substitute, Connect, Query, and Match) to generate an RCL tree Ψ from the sentence "*place the green sphere over the red cube*" from the Dukes (2013) dataset.

5.3.2 Group concepts to generate RCL elements

Once the sentence S is transformed into a list of visual concepts S', we aim to group these concepts to find elements that can be linked back to the input video clip. We group the visual concepts in S' to create all possible *entity, action, spatial-relation*, and *destination* RCL elements. The definitions of these elements were mentioned earlier in Table 5.1 and are assumed to be known to the robot. The grouping of these elements is performed by connecting (*i*) consecutive *colour*, *shape*, and *location* concepts to form *entity* RCL elements; (*ii*) consecutive *action* concepts to form *action* RCL elements; (*iii*) consecutive *direction*, *distance*, and *entity* concepts to form *spatial-relation* RCL elements; and (*iv*) each *spatial-relation* and *location* concept forms a *destination* RCL element. For example, in the sentence $S' = \langle action_1, colour_2, shape_3, direction_1, colour_3, shape_1\rangle$ the concepts $colour_2$ and $shape_3$ are grouped together to generate an *entity* element of the form $entity(colour_2, shape_3)$, i.e. an *entity* element with two children: $colour_2$ and $shape_3$, as shown in the *Connect* section in Figure 5.6. Similarly, the concepts $colour_3$ and $shape_1$ are grouped together to generate another *entity* element. Note that $shape_3$ and $colour_3$ were not grouped together because there is a *direction* concept between them, and our grouping method requires the concepts to be consecutive in the sentence. Also, the ordering and number of concepts are not constrained in the grouping procedure, i.e. an *entity* element can be created by grouping a *colour* concept followed by *shape*, or vice versa. This allows the learning of grammar from different languages where adjectives and nouns are ordered differently. The same grouping procedure applies to *action, spatial-relation* and *destination* elements. For example, each of the two entities mentioned earlier is grouped with the $direction_1$ concept to generate a spatial relation element as shown in the Figure 5.6. By grouping different concepts together, we managed to create different RCL elements and different sentence structures. Each of which tells a different story as to what happened in the input video as will be explained in the following section.

5.3.3 Query RCL elements

The query process aims to link RCL elements found in the previous section to objects and relations in the input video clip. This is achieved by linking each (*i*) *entity* element in the sentence to an object *id*, (*ii*) *location* element to a location concept, and (*iii*) *spatial-relation* element to a relation concept. The linking is enabled by querying the children of RCL elements with the list of predicates extracted from the input video clip. The extraction of predicates was previously mentioned in §Visual World Encoding (section 3.3.4), and the list of predicates for this example are shown in Figure 5.4.

To perform the linking of RCL elements to the input video, we query the children of each element against the encoded predicates from the input video. The aim is to find all objects or relations in the input video that satisfies the constraints imposed by the children of each element. For instance, to link the *entity* element $entity(colour_2, shape_3)$, we query its children: $colour_2$ and $shape_3$ looking for all objects that have both of these properties attributed to them. By inspecting the list of predicates shown in Figure 5.4, we can see that the green sphere, the object with $id = 0$, is the only object that satisfies both constraints. Therefore, the *entity* element $entity(colour_2, shape_3)$ is linked to $id = 0$, and by doing so we successfully linked part of the input sentence to the input video, i.e. the robot now knows that this part of the sentence ("*green sphere*") is describing the green sphere object in the input video. Similarly, querying the *entity* element $entity(colour_3, shape_1)$ will result in linking it to the red cube object with $id = 1$. The same technique applies for destinations. For example, the *destination* element *spatial-relation(direction*$_1$, *entity(colour*$_3$,*shape*$_1$)) returns the spatial relation predicate $direction_1(0, 1)$. This means that the final destination of the manipulated object has to satisfy this spatial relation, meaning the object with id=0 should be located at direction$_1$ with respect to the object with id=1. This is repeated for all found entities and destination elements in the input sentence, as shown in Figure 5.6 (*Query*).

If multiple objects in the scene satisfy a query, a list of *id*s is returned, while if there are none, the query returns an empty list, this might happen due to noise in vision and/or language. In the next section, we match the results found by querying elements with the values of vision tree elements $\{action_v,\ entity_v$ and $destination_v\}$.

5.3.4 Matching RCL elements with Ω

Given the query results of RCL elements, we aim to find the correct language structure Ψ by matching the query results to the elements of the vision tree Ω. I.e. we aim to find the correct language structure that reflects what happened in the input video. This is achieved by comparing the values of each RCL element with the vision elements. For example, the vision tree Ω in Figure 5.5 has an $entity_v$ element with id=0. By matching this with the available RCL elements we find that the $entity(colour_2, shape_3)$ which is describing the green sphere object holds the same object *id*. Therefore, the two are matched together, as shown in the *Matching* section in Figure 5.6. Similarly, the $action_v$ element in the vision tree holds the value $action_1$, which is matched to the available *action* element in the input sentence for the word '*place*'. Finally, the $destination_v$ element in the vision tree holds two potential values: the location concept $location_4$, and the spatial relation $direction_1(0, 1)$. By looping through the available options, we match the spatial relation described by the words '*over the red cube*' with the $destination_v$ element in the vision tree. By performing the matching, the robot now has the correct sentence structure that reflects what happened in the input video, i.e. the robot now has an RCL tree that it can use to learn the grammar rules of natural language. The resultant sentence structure Ψ from this example is shown in the Figure 5.7. The names used in the tree (*colour, shape, spatial-relation, etc.*) are used for simplicity/readability. The robot is not assumed to know these words specifically but knows of the existence of these elements. In the following section, the learning of probabilistic grammar rules using this tree is discussed.

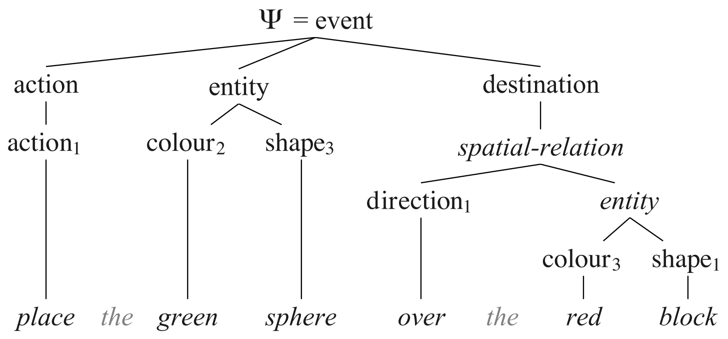


Figure 5.7: The generated RCL tree Ψ from the example shown in Figure 5.4 using the automatic language tree generation algorithm presented in Algorithm 1.

5.4 Learning Grammar Rules

Grammar induction refers to the process of learning a formal grammar usually as a collection of re-write rules or productions from a set of observations. The observations usually consist of natural language sentences annotated with grammar trees. These observations are used to train a parser by learning the grammar rules. In this thesis, Probabilistic Context Free Grammar (PCFG; also known as Stochastic CFG) is used to model the grammar rules of language. The

PCFG is presented in the NLP literature in the form $\Pi = (N, T, R, S, P)$, where Π is the language grammar, N is the set of non-terminal symbols, T is the set of terminal symbols, R is the set of production rules, S is the start symbol, and P is the set of probabilities on production rules. The productions rules are of the form $(left\ handside, right\ handside, weight)$. A production (lhs, rhs, w) is written as $lhs \rightarrow^{w} rhs$, such that $lhs \in N$, $rhs \in N \times T$, and $\forall_i \Sigma_j P(lhs_i \rightarrow^{w} rhs_j) = 1$. The probability w of each rule is proportional to the number of times this rule is observed in the training data. An example of PCFG is shown in Figure 5.8 along with a parsed tree for a sentence using this grammar.

In this thesis, we show how we learn PCFG rules by mapping natural language commands to visual features extracted/learned from input video clips. The main contribution in our grammar induction approach is that we automatically generate training examples similar to those annotated by human experts shown in Figure 5.1 as presented in the previous section. The generation of such training data is achieved by employing three different components obtained from linguistic and visual inputs, which are: (1) the learned visual concepts presented in Chapter 3, (2) the learned language groundings presented in Chapter 4, and (3) the extracted vision trees presented in the previous section. By combining all thee of them we successfully replace the human expert annotations of RCL trees, enabling the robot to learn about natural language grammar without human supervision.

T = {with, saw, astronomers, ears, stars, telescopes}
N = {S, PP, P, NP, VP, V}
S = {S}

R =	S $\xrightarrow{1.0}$ NP VP	NP $\xrightarrow{0.18}$ stars
	PP $\xrightarrow{1.0}$ P NP	NP $\xrightarrow{0.04}$ saw
	VP $\xrightarrow{0.7}$ V NP	NP $\xrightarrow{0.4}$ NP PP
	VP $\xrightarrow{0.3}$ VP PP	NP $\xrightarrow{0.1}$ astronomers
	P $\xrightarrow{1.0}$ with	NP $\xrightarrow{0.18}$ ears
	V $\xrightarrow{1.0}$ saw	NP $\xrightarrow{0.1}$ telescopes

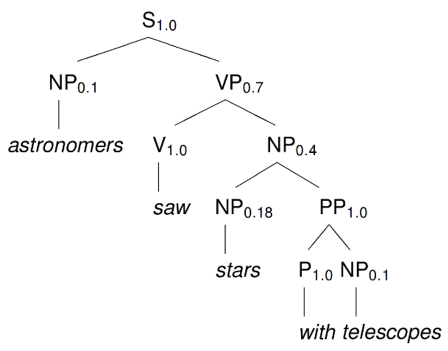


Figure 5.8: PCFG example. (left) A probabilistic context free grammar $\Pi = (N, T, R, S, P)$. The probability of each rule is shown on the right side of each arrow. (right) A parsed tree for the sentence "*astronomers saw stars with telescopes*" using the grammar Π.

5.4.1 Learning a PCFG Π from RCL tree Ψ

To provide our robot with the ability of understanding new natural language commands, we learn/induce a grammar $\Pi = (N, T, R, S, P)$ from the automatically generated language tree (Ψ) shown in Figure 5.7. The induced grammar rules are used to parse new commands into RCL trees that the robot can understand and execute. The grammar rules are modelled using Probabilistic Context Free Grammar (PCFG). To learn the grammar rules, we follow the Inside-Outside algorithm presented by Lari and Young (1990). To induce a PCFG grammar rule, i.e. learn the probability of a production from a list of observations we use Equations 5.1 and 5.2. There are only two kinds of productions in the grammar rules we learn: the non-terminal ones ($\mathcal{B} \to \mathcal{C}_1, \ldots, \mathcal{C}_m$), and the terminal ones ($\mathcal{B} \to \mathcal{Z}$), where $\mathcal{B}$ and $\mathcal{C}_i$ are non-terminal symbols, while $\mathcal{Z}$ is a terminal symbol. A probability, called $P(\mathcal{C}_1, \ldots, \mathcal{C}_m|\mathcal{B})$ or $P(\mathcal{Z}|\mathcal{B})$, is associated to each production. The computation of these probabilities are shown in Equations 5.1 and 5.2, where P is the probability of the grammar rule, λ is a counting function, and $*$ is any right hand side for the grammar rule, i.e. any grammar rules with a left hand side $\mathcal{B}$. A normalization condition must hold for every non-terminal $\mathcal{B}$, which is the summation of the probabilities of all rules where $\mathcal{B}$ is the left side of it must equal to one, as shown in Equation 5.3.

$$P(\mathcal{C}_1, \ldots, \mathcal{C}_m|\mathcal{B}) = \frac{\lambda(\mathcal{B} \to \mathcal{C}_1, \ldots, \mathcal{C}_m)}{\lambda(\mathcal{B} \to *)} \tag{5.1}$$

$$P(\mathcal{Z}|\mathcal{B}) = \frac{\lambda(\mathcal{B} \to \mathcal{Z})}{\lambda(\mathcal{B} \to *)} \tag{5.2}$$

$$\sum_{c} P(c_1, \ldots, c_m|\mathcal{B}) + \sum_{z} P(z|\mathcal{B}) = 1 \tag{5.3}$$

To learn the grammar rules we start with an empty Probabilistic Context Free Grammar (PCFG) rule set. The rules learned from the example sentence "*place the green sphere over the red cube*" are shown in Table 5.2. The rules learned from all examples are accumulated into one PCFG grammar Π. These rules model the structure of natural language commands and are used to parse new commands into RCL trees which the robot can understand and execute.

Learning Grammar Rules	
Grammar Rules	*Probabilities*
event → action, entity, destination	1.0
entity → colour, shape	1.0
destination → spatial-relation	1.0
spatial-relation → direction, entity	1.0
action → *place*	1.0
direction → *over*	1.0
shape → *sphere*	0.5
shape → *cube*	0.5
colour → *green*	0.5
colour → *red*	0.5

Table 5.2: The learned grammar rules from the example sentence "*place the green sphere over the red cube*" are shown on the left side, while the probability of each rule is shown to the right.

5.5 Discussion

In this section we discuss the main contributions presented in this chapter, along with the assumptions made to enable the learning of grammar rules and the limitations of our approach.

5.5.1 Main contributions

In this chapter, a new probabilistic grammar induction approach of natural language commands was presented. The learning was achieved in a semi-supervised manner using language and vision inputs. The learning of grammar rules from unlabelled linguistic inputs was enabled by matching the language structure with the vision tree of the input video. To the best of our knowledge, this work is the first to learn grammar rules using extracted/learned visual concepts and language groundings. The automatic generation of training data similar to the ones annotated by a human expert is the main contribution we offer in this chapter.

5.5.2 Assumptions

The video and sentence each have to contain only one action, the RCL elements have to be known before hand, and the matching has to be complete between the vision and RCL trees.

5.5.3 Limitations

Even though our grammar induction approach can be expanded to more domains such as robot navigation or cooking recipes, it is not an easy task to do so. More RCL elements would have to be manually defined and provided to the robot, which makes it less suitable to learn from different domains at once. However, we believe that this grammar induction approach takes a step closer towards building a system that can autonomously generate new RCL elements and learn in an unsupervised manner the grammar rules of natural language by connecting language to vision. Also, we do not use the gained knowledge to improve the chances of learning a new grammatical form, or learn the meaning of a new word, which remains as future work.

Chapter 6

Experimental Procedure

"If a machine is expected to be infallible, it cannot also be intelligent."

—*Alan Turing*

In this thesis, a novel, loosely-supervised and incremental learning framework is presented that enables robots of bootstrapping their knowledge in language and vision domains. The framework contains a number of existing and newly developed machine learning techniques that focus on learning three aspects of language and vision, (a) visual concepts, (b) language groundings and (c) grammar rules. In this chapter, we evaluate each of the presented machine learning techniques in this thesis, and compare them against other supervised and unsupervised systems when applicable.

This chapter is organised as follows. First, we list the experiments used to evaluate our learning framework along with the evaluation measures used in this thesis. Second, we list the robots and collected datasets used in evaluating our learning framework. Third, we describe in detail each of the experiments and present the results of each.

6.1 Experiments and Evaluation Measures

The performance of the presented language and vision learning framework is evaluated using four experiments that are designed to evaluate the framework's ability in:

1. Incremental learning of visual concepts from video inputs. This experiment evaluates the use of Incremental Gaussian Mixture Models (IGMM) technique along with a Bayesian Information Criterion (BIC) for simple visual concepts learning such as faces, distances, shapes, etc., and the use of spatio-temporal DAGs along with LDA and Variational Bayes algorithm for complex concepts learning such as robot and human actions.

2. Incremental language groundings of n-grams to visual concepts. This experiment evaluates the performance of the integer programming technique in correctly finding the correct multi-to-multi mappings between language and vision compared to other methods like the supervised Hidden Markov Model (HMM) for Part-of-Speech (POS) tagging system which require the ground truth groundings to learn from in a supervised manner.

3. Incremental grammar rules induction. This experiment evaluates the incremental learning of probabilistic context free grammar (PCFG) rules from pairs of inputs, where each pair contains a video clip along with its corresponding natural language command describing it. The learned grammar rules are evaluated based on their ability to correctly parse new (previously unseen in training data) natural language commands into Robot Control Language (RCL) trees which our robots can understand and execute.

4. Scalability of the learning framework. This experiment briefly evaluates how the learning framework scales with large amounts of data by presenting the memory requirements of our learned models in an incremental manner compared with the size of the raw data.

A number of evaluation measures are used to evaluate the performance of our system in each experiment. The measures along with their descriptions are presented in the following sections.

6.1.1 F_1 score

In binary classification statistical analysis, F_1 score (also known as the F-measure or F-score) is a measure of the accuracy of a binary classification test. As presented by Rijsbergen (1979), the F_1 score is computed using both the *Precision* and *Recall* of the test, where *Precision* is the number of correct positive results divided by the number of all positive results, and *Recall* is the number of correct positive results divided by the number of total relevant positive results. The

F_1 score can be thought of as a weighted average of the precision and recall of a classification test, where it is equal to 1 at its best value (every point in the test classified correctly) and 0 at its worst. The F_1 score is computed using Equation 6.1.

$$F_1 = 2 \times \frac{1}{\frac{1}{precision} + \frac{1}{recall}} = 2 \times \frac{precision \times recall}{precision + recall} \tag{6.1}$$

6.1.2 V measure

The V-measure (V_m) presented by Rosenberg and Hirschberg (2007) is a combination of both 1-*homogeneity*: a metric of cluster labelling given the ground truth classes, it measures whether each predicted cluster contains same-class data points as shown in Equation 6.2, where C is the ground truth classes, K is the clusters and $H()$ is the entropy. 2-*completeness*: a metric measuring whether all data points that are members of a given class are elements of the same predicted cluster measured using Equation 6.3. The V-measure metric provides a measure of similarity of any two sets of class labels, where 0 indicates no correlation and 1 indicates perfect correlation, and is computed using Equation 6.4.

$$homogeneity = 1 - \frac{H(C|K)}{H(C)} \tag{6.2}$$

$$completeness = 1 - \frac{H(K|C)}{H(K)} \tag{6.3}$$

$$V_m = 2 \times \frac{homogeneity \times completeness}{homogeneity + completeness} \tag{6.4}$$

6.1.3 Normalised Mutual Information (NMI)

The Mutual Information (MI) between two sets of labels is computed using Equation 6.5, where $P(x, y)$ is the joint probability distribution of X and Y, and $P(x)$ and $P(y)$ are the marginal probability distribution functions for X and Y as presented by Cover and Thomas (1991). This metric provides a measure of similarity of any two sets of class labels. The Normalised Mutual Information (NMI) provides a normalised measure of MI as shown in Equation 6.6, where H represents the entropy of the set. The NMI provides an output of 0 to indicate no mutual information between the sets X and Y, and 1 to indicate perfect correlation.

$$\mathrm{MI}(X,Y) = \sum_{y\in Y}\sum_{x\in X} P(x,y)\log\left(\frac{P(x,y)}{P(x)\ P(y)}\right) \tag{6.5}$$

$$\mathrm{NMI}(X,Y) = \frac{\mathrm{MI}(X,Y)}{\sqrt{H(X)\ H(Y)}} \tag{6.6}$$

6.2 Datasets

Four different datasets are collected/extended and used to evaluate the performance of our language and vision learning framework. The datasets include three robot manipulators performing different table-top tasks such as picking up and moving blocks, and one mobile robot observing humans performing various kitchen activities such as making tea or microwaving food. The four datasets are presented in more detail in the following sections.

6.2.1 Extended Train-Robots Dataset

Extended Train-Robots is a simulation dataset with a 3 DOF robot arm along with a two fingered gripper performing various table-top manipulation tasks in a simulated block world environment. This dataset is an extended version of the Train-Robots dataset presented by Dukes (2013). The Train-Robots dataset aims to improve natural language human-robot spatial interaction through verbal commands. It was designed to develop systems capable of understanding natural language commands with spatial information on an 8×8 chessboard simulated world. The dataset contains 1000 scenes, where each scene consists of two images. One represents the initial configuration of the world, and the second represents the desired (or final) one. In each scene, only one object changes its location as shown in Figure 6.1. After the scenes were generated, non-experts were asked to annotate the 1000 scenes with appropriate natural language commands such that if these commands were given to a robot, the robot would be able to change the scene from the initial to the desired configuration. An example to these commands for the scene shown in Figure 6.1 is "*move the yellow prism to on top of the grey tower*". Amazon Mechanical Turk (2010) was used to collect the natural language commands, 4850 commands were collected and annotated with appropriate Robot Control Language (RCL) trees as shown in Figure 6.2. The original *Train-Robots* dataset contains two shapes only (cube and prism), and eight different colours (red, green, blue, cyan, grey, white, yellow and pink).

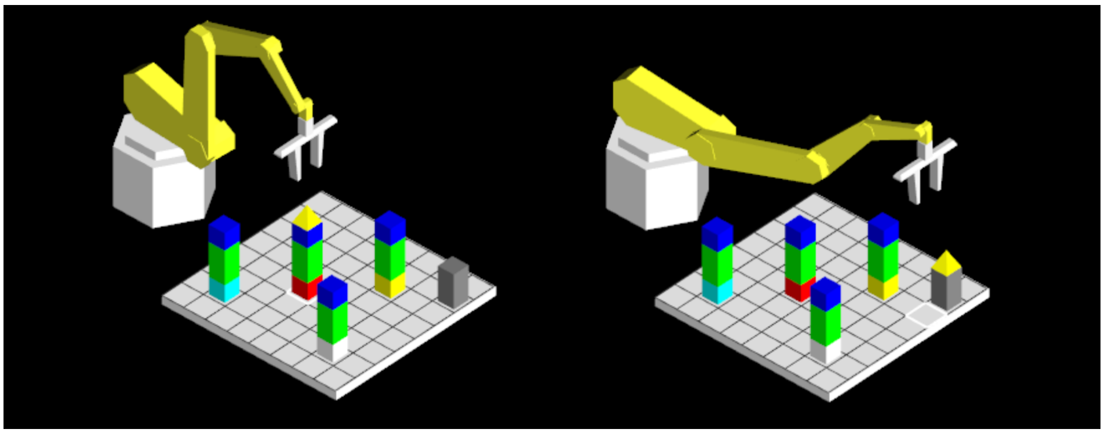


Figure 6.1: A scene example from the Train-Robots dataset with two images, the initial configuration (left) and the desired or final configuration (right). Image taken from Dukes (2013).

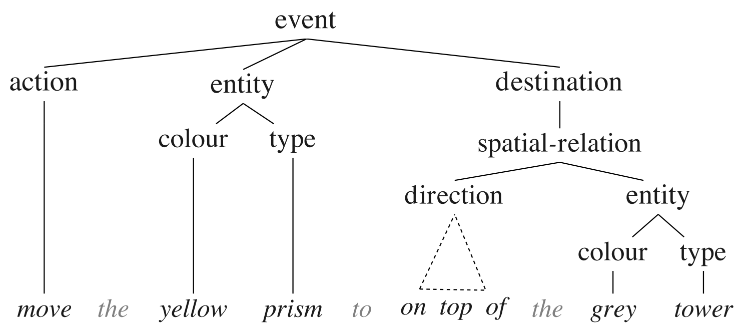


Figure 6.2: An Example of a human annotated RCL tree from the Train-Robots dataset.

In this work, the Train-Robots dataset[1] is extended in a number of ways. First, the dataset contained only two shapes one of which (the cube) existed in almost every scene, also the red colour existed in every scene. This limits the learning ability of our robot as it will associate every word in the input sentences with the cube shape and the red colour. Therefore, we modified the scenes to include two more objects (*sphere, cylinder*) and one more colour (*black*). This is achieved by changing half the scenes that contains *prisms* to *spheres*, *cubes* to *cylinders*, and *red* to *black*. The scenes were randomly selected and were different for changing the prisms, cubes and red. Out of the 1000 scenes, 500 were randomly chosen to change prisms to spheres,

[1]The original and extended versions of the dataset are available at http://doi.org/10.5518/32

and a different 500 were chosen to change cubes to cylinders, etc. Particular care was taken in modifying the annotated natural language commands to match the scenes in order not to alter the meaning or any mistakes in the descriptions. This allows the dataset to be more rich and thus contains more variation to learn from. The second extension to the *Train-Robots* dataset was to automatically animate the 1000 scenes to produce videos of the robot performing the action. Examples of key frames for the generated videos are shown in Figure 6.3. The third extension is the translation of all 4485 commands from English to Arabic to test the learning framework on a different language. The translation was performed using Goslate (2016), a free Google translator API for the Python programming language. Again, particular care was taken not to alter the commands or correct any mistakes before translation.

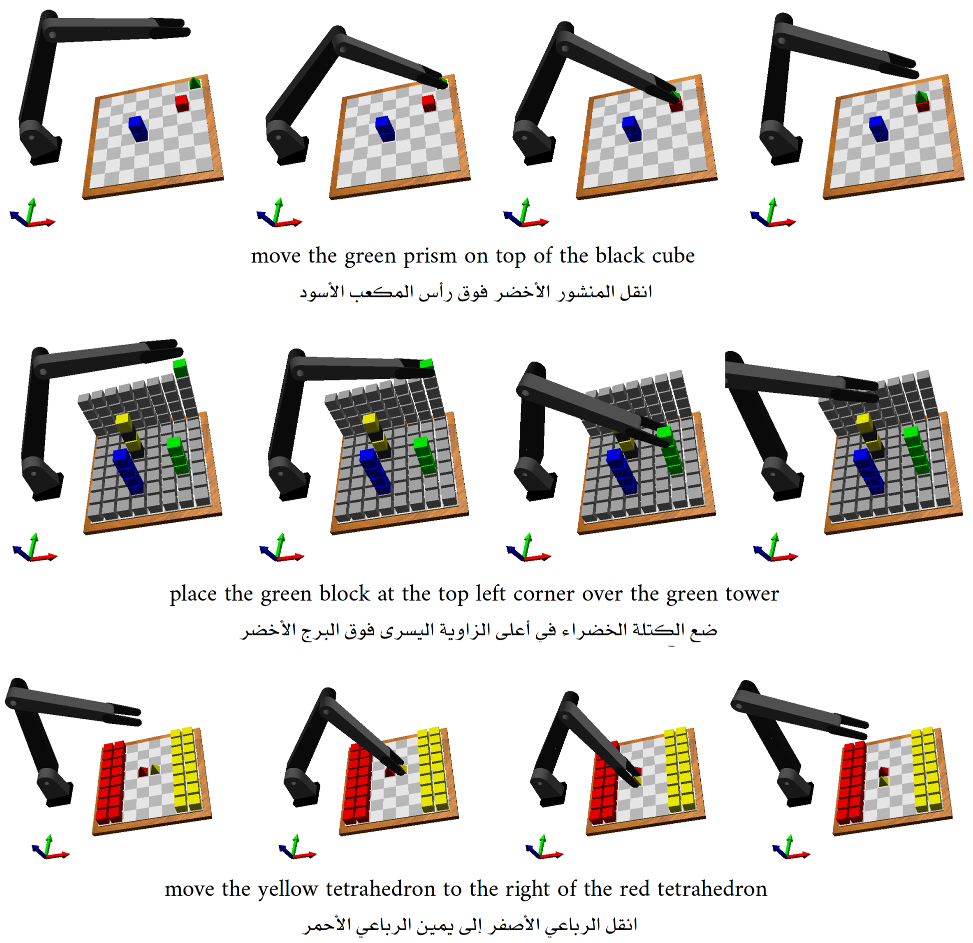

move the green prism on top of the black cube
انقل المنشور الأخضر فوق رأس المكعب الأسود
place the green block at the top left corner over the green tower
ضع الِكتلة الخضراء في أعلى الزاوية اليسرى فوق البرج الأخضر
move the yellow tetrahedron to the right of the red tetrahedron
انقل الرباعي الأصفر إلى يمين الرباعي الأحمر

Figure 6.3: Examples from the Extended Train-Robots dataset along with their annotated commands; the Arabic sentences are translated from the English ones.

6.2.2 Leeds Robotic Commands Dataset

The Leeds Robotic Commands dataset[2] was first presented in Alomari *et al.* (2017b). The dataset contains real-world RGB-D scenes of a robot manipulating different objects together with natural language descriptions of these actions. For this dataset, we used a Baxter robot (LUCAS) from Rethink Robotics as the robotic platform. We fitted LUCAS with a Microsoft Kinect2 sensor (2015) on its chest such that it can observe and model its environment in RGB-D as shown in Figure 6.4. The Kinect2 was used to collect RGB-D videos of LUCAS performing various manipulation tasks with real objects from the robot's point of view. To record each video, an annotator was asked to perform a given task by driving LUCAS using a joystick[3]. In each video, only one object is manipulated (i.e. only one object changes its location). The three commands used in this dataset to guide the annotators are '*pick up*', '*put down*' and '*move*'. For example, the command '*move the red apple into the white bowl*' was provided to an annotator who guided LUCAS to perform the action as shown in Figure 6.5.

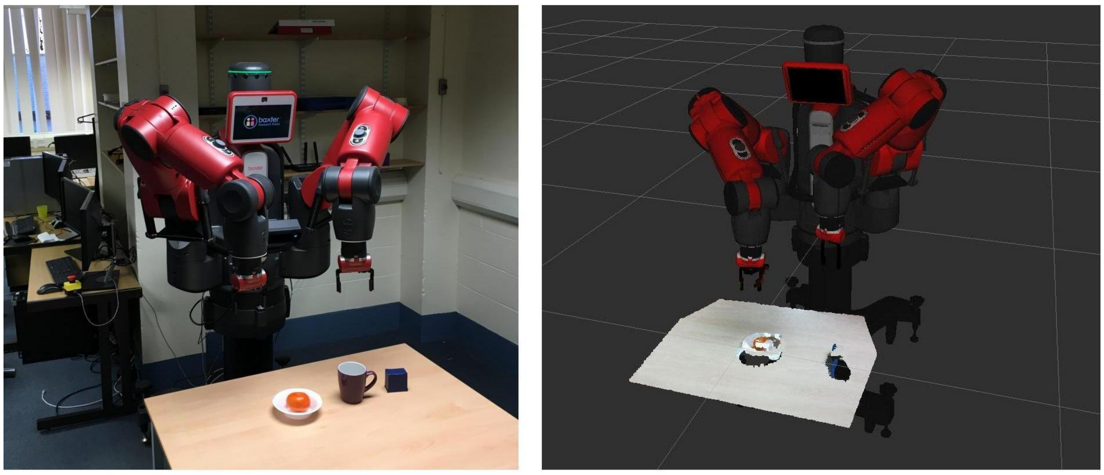


Figure 6.4: Leeds Robotic Commands dataset setup. (left) The Baxter robot (LUCAS) used as the robotic platform in this dataset, and is fitted with Microsoft Kinect2 sensor on its chest to record data. (right) The point-cloud generated from the Kinect2 sensor after calibrating its position with respect to LUCAS's body frame.

The dataset includes 204 video clips consisting of 17,373 frames in total of LUCAS manipu-

[2]The Leeds Robotic Commands dataset is available at http://doi.org/10.5518/110

[3]The Python and ROS implementation of the joystick commands and their mapping to the velocity control of LUCAS's arms are available at https://github.com/OMARI1988/baxter_pykdl

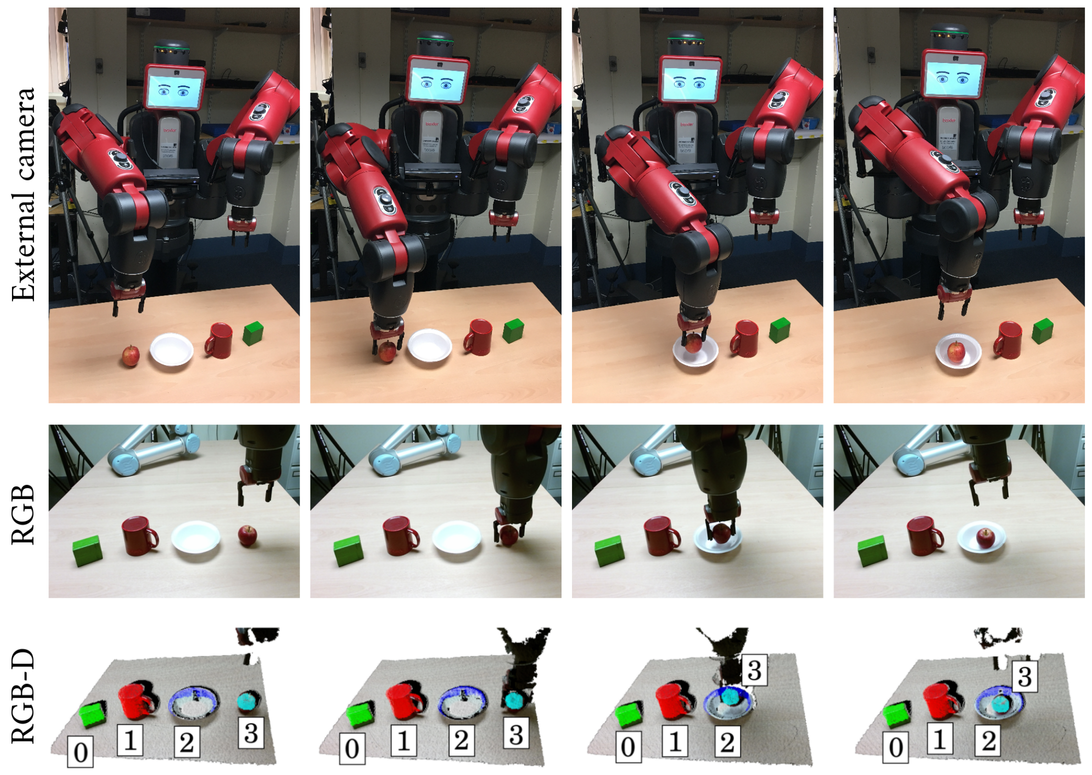


Figure 6.5: Example from the Leeds Robotic Commands dataset for the command '*move the the red apple into the white bowl*'. (top) An external camera is placed opposite the robot to record the scene. Note that this camera is not used in objects detection nor tracking. (middle) The RGB feed from the Kinect2 sensor showing the point-of-view of the robot. (bottom) The RGB-D feed from Kinect2, along with the detected objects' *id*s and tracks.

lating various objects. A total of 51 different objects are manipulated in the dataset that include basic block shapes, fruits, cutlery, and office supplies, with an average of five objects present in each scene. The objects are detected using an off-the-shelf table-top object detector presented by Muja and Ciocarlie (2013) which I implemented in the Robotics Operating System (ROS). This technique is used to drive the attention of the robot to the graspable objects placed on a table within the robot's reach. Examples of detected objects are shown in Figure 6.6. Once an object is detected in a video clip, the location of this object is tracked across all remaining frames using a six dimensional particle filter presented by Klank *et al.* (2009) which has a C++ implementation in the Point-Cloud-Library (PCL 2011). The six dimensions of the particle filter are the three x, y, z location, and the three r, g, b colour values of each pixel in the object segment. The object detection and tracking techniques are also shown in the bottom row of

Figure 6.5 where each detected object is assigned a unique *id* and tracked throughout the video.

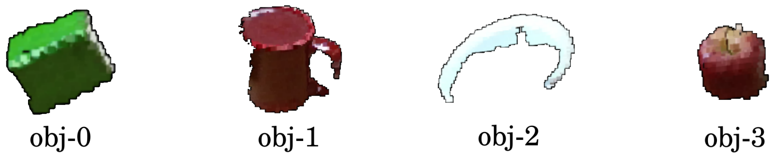


Figure 6.6: Table-top object detection technique for the example shown in Figure 6.5.

For the linguistic domain, the videos were annotated with appropriate natural language commands by a separate group of annotators. The annotators were presented with the video clips, one at a time, and were asked to provide appropriate natural language commands for each clip in such a way that if the command was provided to LUCAS, then it would be able to perform the command with no ambiguity. The dataset contains a total of 1024 natural language commands describing the 204 videos, an average of five per video. Examples from the Leeds Robotic Commands dataset showing the collected videos and the annotated linguistic commands are presented in Figure 6.7.

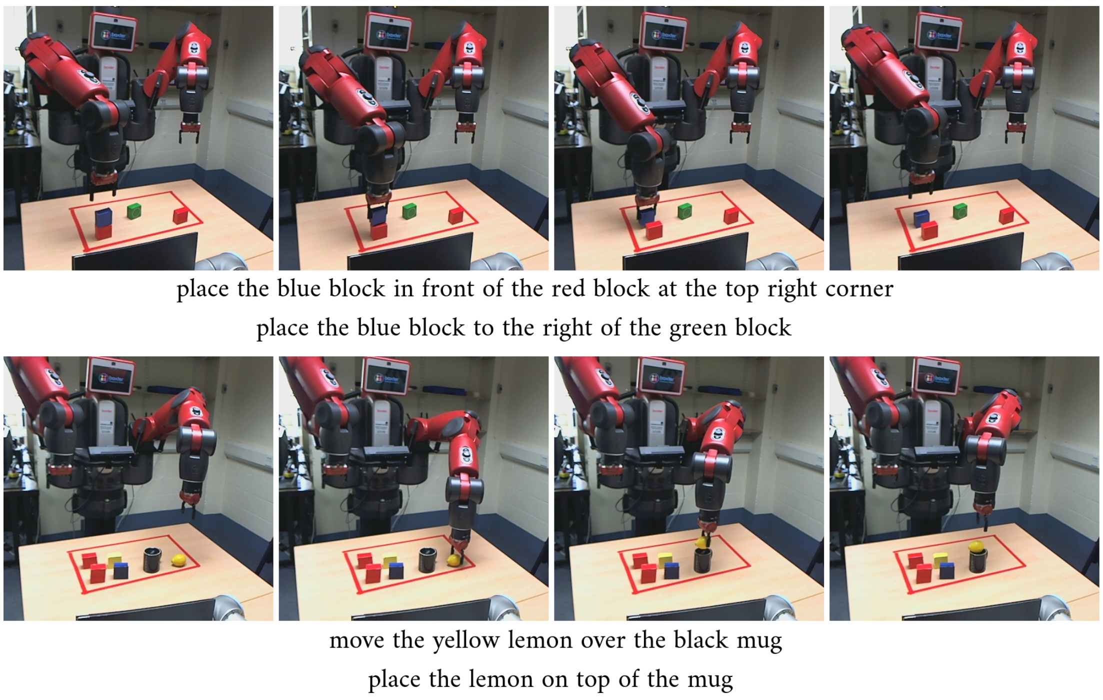

place the blue block in front of the red block at the top right corner
place the blue block to the right of the green block
move the yellow lemon over the black mug
place the lemon on top of the mug

Figure 6.7: Two examples from the Leeds Robotic Commands dataset along with examples from their annotated natural language commands.

6.2.3 Extended Object Ordering Dataset

The original Object Ordering dataset was presented by Sinapov *et al.* (2016). The robot used in this dataset is a custom built mobile manipulator that uses the Segway Robotic Mobility platform (2004) along with a 6-DOF Kinova Mico arm (2014) with a two fingered gripper as its end effector. The robot is shown on Figure 6.8 (left) along with all the objects used in this dataset. The Object Ordering dataset was originally designed to teach a robot how to arrange objects in an ascending order based on their properties. For example, to arrange objects from shortest to tallest, smallest to largest, lightest to heaviest, etc. To learn about object properties, the robot performs seven different predefined actions on each object in the scene. The predefined actions are *grasp, lift, lower, drop, press, push* and *hold*, these actions are shown in Figure 6.8 (right).

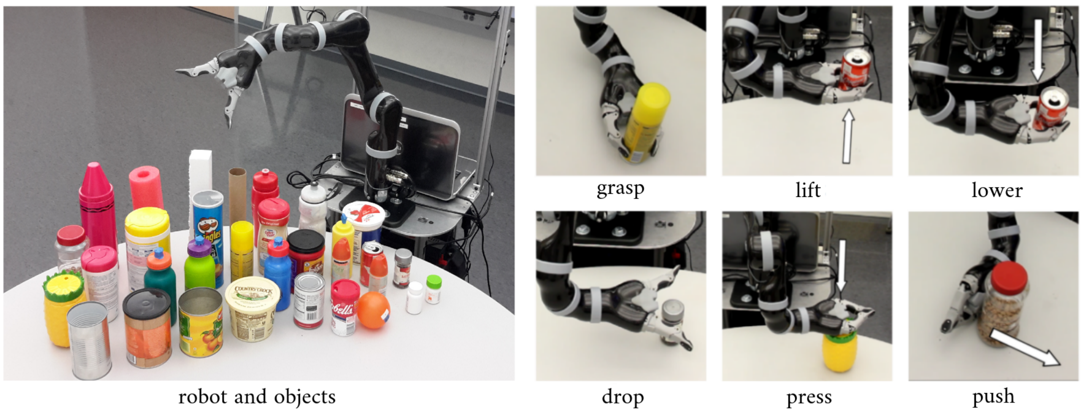


Figure 6.8: The Object Ordering dataset robot. (left) The Segway Robotic Mobility and the 6-DOF Kinova Mico arm mobile manipulator along with all the objects used in this dataset. (right) Six of the seven predefined actions used to teach the robot about object properties. Both figures are copied from Sinapov *et al.* (2016).

The set of objects that the robot explores and learns about consists of 32 common household items including cups, bottles, cans, and other containers. The object properties varied in weight, height, and width. The objects' height and width was measured in millimeters while their weight was measured in grams in this dataset. The objects were chosen in this dataset such that the distributions of their weight, width, and height were roughly uniform. Also, each video clip features only a single object in it, which means the robot cannot learn about spatial relations between objects in this dataset.

We extended the Object Ordering dataset by annotating the video clips with appropriate natural language commands. The commands were provided by annotators who viewed the video clips, one at a time, and wrote appropriate commands such that if the command is provided to the robot, the robot would be able to perform the task with no ambiguity. The dataset contains a total of 1120 video clips, which were annotated with 1120 linguistic commands, one for each clip. Examples of video clips and the their corresponding commands are shown in Figure 6.9.

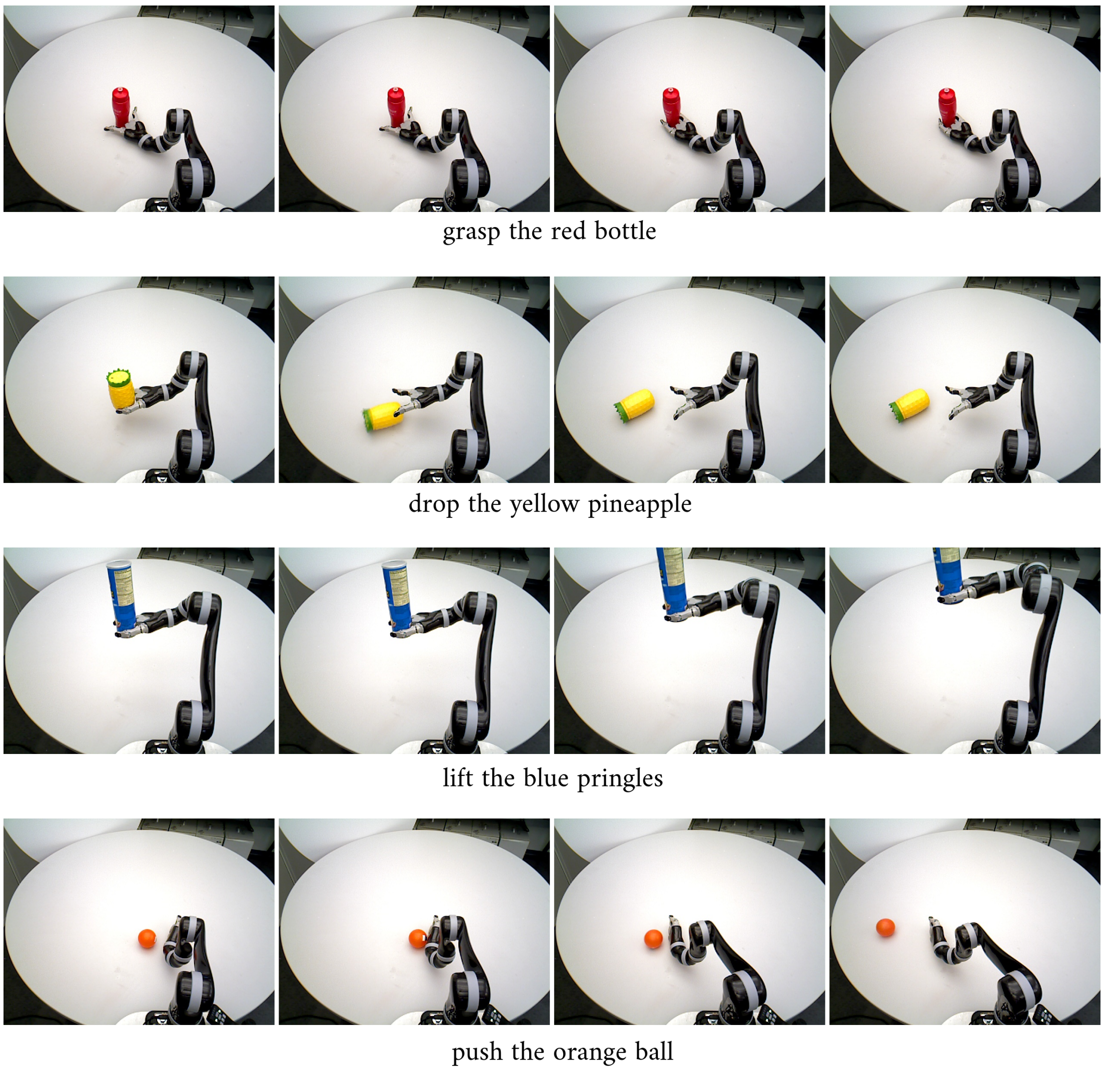


Figure 6.9: Four examples from the extended Object Ordering dataset along with their annotated natural language commands.

6.2.4 Extended Kitchen Activities Dataset (LUCIE)

To integrate into human environments, mobile robots with collaborative human-oriented tasks should be enabled to continuously learn about their environments, the people who inhabit these environments, and the activities that take place there. From an autonomous robot point of view, this requires incremental learning methods that operate on the outputs of various kinds of sensor modalities the robot might have, ranging from laser rangefinder and RGB-D cameras to voice recognition. The desired outcome of this process is learning a collection of grounded concepts of the robot's environment that are beneficial for the robot's specific task.

In this dataset, we present a demonstration of our learning framework for symbol grounding for autonomously-extracted components of real-world, human environments for a mobile robot. The novelty of our framework is that it extends existing work in autonomous symbol grounding towards 'the wild' from the typical lab settings towards more realistic, real-world scenarios, and from ideal sensing conditions to noisy, limited and changing perception of a mobile robot. Moreover, it does this in a loosely-supervised, incremental fashion. We presuppose that the robot can navigate and visually analyse the environment to extract a multitude of visual features in order to incrementally recover useful visual concepts. If natural language descriptions of the observations are also provided, they can be analysed along with the visual features to ground the words describing people, objects, activities, etc. to their most relevant perceptual concepts. One possible application of such a framework could be in the field of security or assistive robotics where robots need the ability to learn on-the-go how to describe new objects or situations in a human-understandable form in a lifelong setting.

For its basic operations, LUCIE (a Metralabs Scitos A5 robot) is equipped with a base-mounted laser scanner used to model the physical environment as a 2D occupancy grid where occupied cells indicate static objects, allowing localisation, mapping and navigation, as shown in Figure 6.10. For this purpose, an off-the-shelf ROS-packages developed by the STRANDS project consortium (2016) is used. Also, the robot is equipped with two RGB-D sensors, one over-head and one chest-mounted, that allow collecting 640x480 RGB video streams in addition to depth point clouds. These sensors are used to generate a 3D map of the robot's environment to search for objects and detect/track humans as they pass within its field of view.

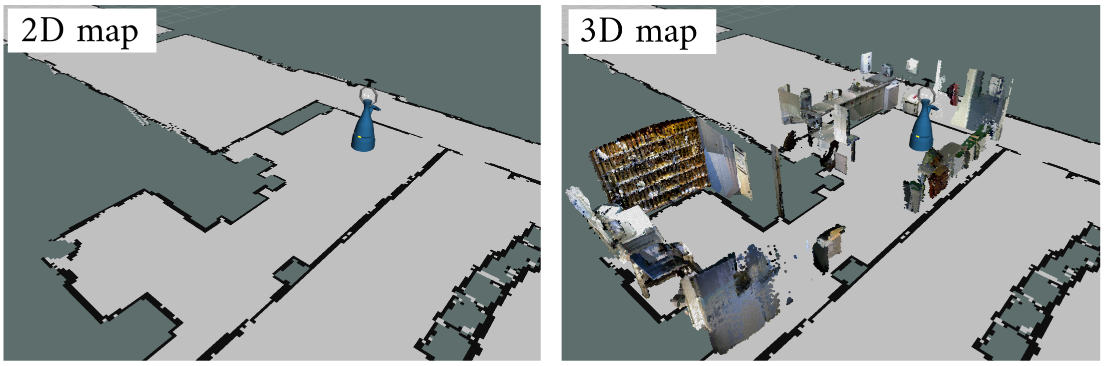


Figure 6.10: Part of the generated map of level-9 in the School of Computing, University of Leeds. (left) 2D map generated with SLAM algorithm using base-mounted laser scanner. (right) 3D map generated by integrating RGB-D scans from the head-mount *xtion* sensor.

For human detection and tracking, the mobile robot LUCIE detects and tracks humans as they pass within the field of view of its head-mounted RGB-D sensor as previously describe in §Human Pose Estimation (3.2.1). The human pose is defined as the estimated 3D position of the person's 15 body joint locations at a single frame in a video clip. The 15 body joints are the *head, neck, torso, shoulders, elbows, hands, hips, knees* and *feet.* For each body joint j, an (xyz) Cartesian coordinate is inferred, and a *human pose estimate* comprises of 15 such joints $J = [j_1, j_2, \ldots, j_{15}]$. To estimate the human pose, a real-time depth-only tracker built on OpenNI (2016) is used along with a post-processing state-of-the-art human pose estimation technique that uses a convolutional pose machine (CPM) by Wei *et al.* (2016) which I integrated with Python and ROS[4]. For each human detected by the robot, a sequence of human pose estimates over a time series of frames is acquired and recorded along with the RGB and depth frames, e.g. Figure 6.11 shows frames from a recorded video clip along with the human pose estimates from both OpenNI and CPM techniques overlayed on the images.

For object detection, LUCIE constructs a 3D model of its environment by fusing RGB-D images into *surfels*, from which the robot generates segments of "objects of interest" as previously described in §Object detection and tracking (3.2.2). In this work, a similar method to that presented by Bore *et al.* (2017) is used, which first splits the scene into a collection of *supervoxels* over which an adjacency graph is formed. Then, weights are assigned to the edges based on local convexity of the point cloud and colour differences between segments. Finally,

[4]The code is available at https://github.com/OMARI1988/cpm_skeleton

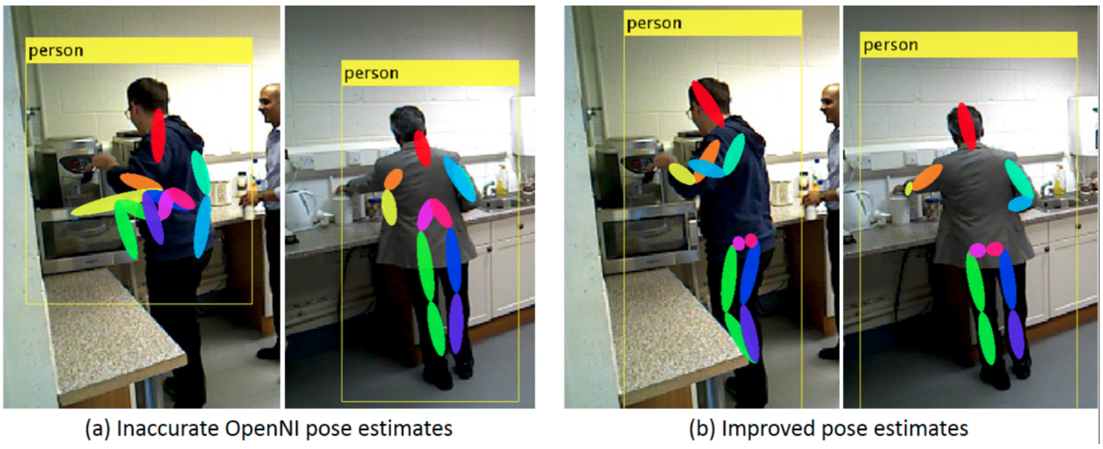

person
person
(a) Inaccurate OpenNI pose estimates
person
person
(b) Improved pose estimates

Figure 6.11: OpenNI and CPM human pose estimation. (left) OpenNI human pose estimation technique is prone to error when body joints are partially occluded. (right) The use of CPM technique to improve the results of OpenNI. CPM finds a better human pose estimate even when joints are partially or fully occluded. Image copied from Duckworth *et al.* (2017).

to segment the point cloud, iterative graph cuts are performed to separate parts with concave boundaries and/or large colour differences. This results in a collection of point cloud segments or objects of interest as illustrated in Figure 6.12. It is important to concentrate attention on the objects that are part of the observed human activities. First, walls, floors and ceilings are removed from the list of objects of interest using a threshold on size and height. Second, the trajectories in 3D space of people in the environment are analysed to extract the locations where people stop more frequently. The objects are scored according to their proximity to people's hands in these locations. The highest scoring objects are considered as the only objects in the environment as presented in Alomari *et al.* (2017a). Examples on these objects are shown in Figure 6.12 (bottom).

In this thesis, we use and extend a publicly available long-term human activity dataset[5] collected over a one week period by our mobile robot LUCIE from multiple view points. The dataset contains 493 video clips each containing a single human performing a simple activity in a kitchen area of an office environment, the activities include, for example, heating food, preparing hot drinks, using a multi-function printer, throwing trash and washing up, amongst others. The length of the videos range between 6 and 2561 frames (as detected by LUCIE's head cam), with a median of 137 frames. On top of the dataset, we collected natural language

[5]The original and extended versions of the dataset are available at http://doi.org/10.5518/86

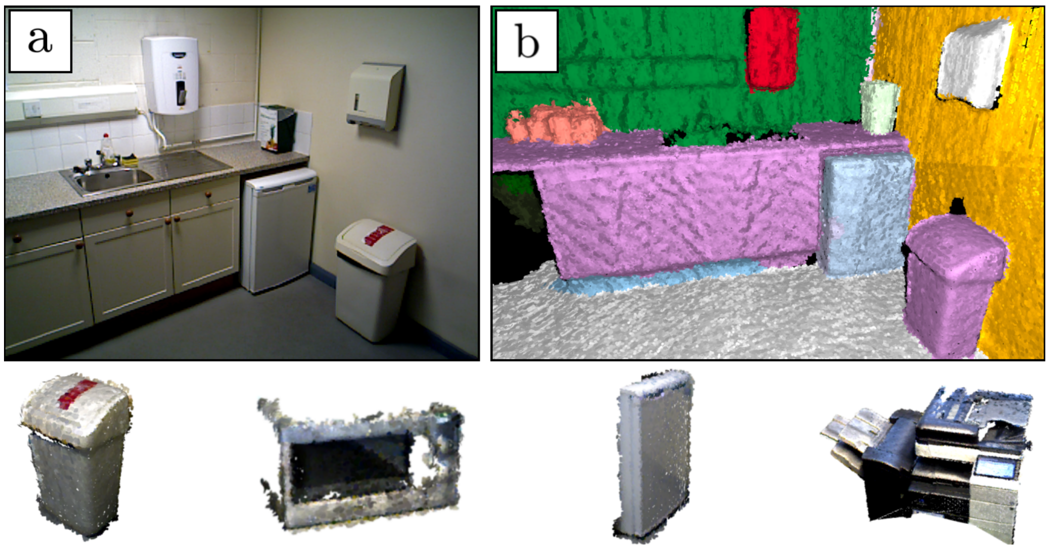


Figure 6.12: The environment observations are fused into a 3D map and segmented. (a) RGB image of the scene, (b) segmented surfel map, or the segmented objects in the scene. (bottom) Examples on objects of interest found after filtering the objects with human trajectories. The objects are from left to right, bin, microwave, fridge and printer.

descriptions of each video clip using Amazon Mechanical Turk, where we requested 'turkers' to describe the activity in the clip and the person's appearance (given a fabricated name). A total of almost 3000 descriptions were collected (6 per clip on average). Example video clips are shown in Figure 6.13 along with a subset of the descriptions obtained.

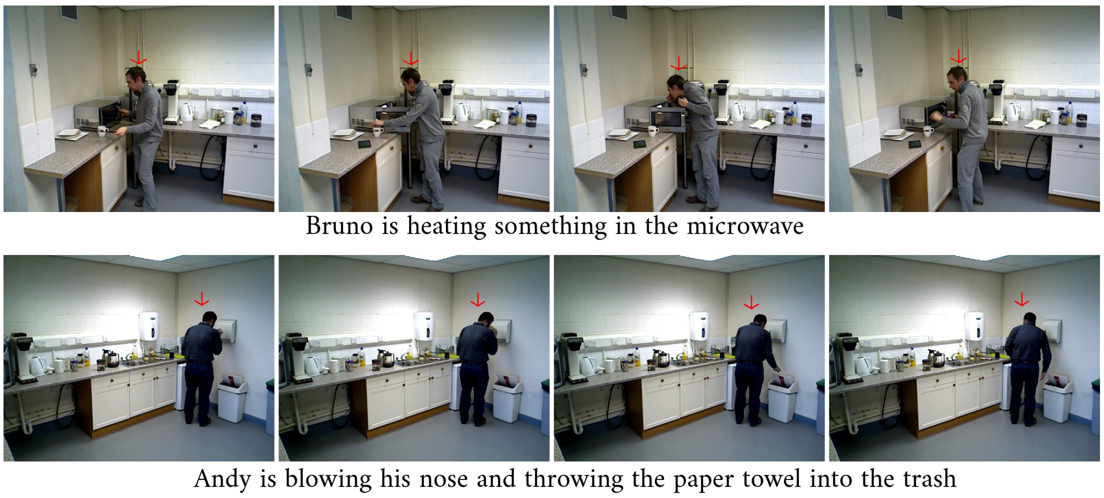

Bruno is heating something in the microwave
Andy is blowing his nose and throwing the paper towel into the trash

Figure 6.13: Two examples from the extended Kitchen Activity dataset along with their annotated natural language descriptions.

6.2.5 Datasets summary

The proposed learning framework for bootstrapping robots' knowledge in language and vision is evaluated against four publicly available datasets, three of which are collected using robot manipulators, and one using a mobile robot. The collected videos and linguistic command numbers are shown in Table 6.1. For example, the Train Robots dataset (in the first row) contains 1000 video clips, with a total of 30,000 frames, 4850 natural language commands describing the 1000 videos, 277 unique words in the 4850 commands, and around 25 objects on average are present in each video clip. The datasets share the predefined visual feature spaces mentioned in §Visual Concepts (Chapter 3), but none of the datasets contains all of them. A summary of all visual feature spaces and their ground truth concepts are shown in Table 6.2. For example, the Train Robots dataset contains nine unique colour visual concepts which are *red, green, blue, cyan, grey, white, yellow, pink* and *black*, that we expect the robot to learn/model using the IGMM technique, and also learn the words used to describe them in natural language as will be shown in the following sections through the different experiments.

Datasets summary - data analysis					
Feature	video clips	frames	commands	unique words	objects (average)
Train Robots	1000	30,000	4850	277	24.8
Robotic Commands	204	17,373	1024	87	5.3
Object Ordering	1120	145,573	1120	31	1
Kitchen Activities	493	67,541	3000	641	5.7

Table 6.1: Data analysis for the four collected datasets. The table shows the number of video clips, frames, annotated commands, unique words in all commands, and average number of objects present in the individual video clips for all four datasets.

Datasets summary - visual concepts							
Feature	*Colour*	*people*	*Object*	*Location*	*Direction*	*Distance*	*Action*
Train Robots	9	–	4	4	5	–	3
Robotic Commands	10	–	13	6	5	6	3
Object Ordering	7	–	6	–	–	–	7
Kitchen Activities	9	17	12	–	–	–	11

Table 6.2: Number of unique visual concepts in *colour, people, shape, location, direction, distance,* and *action* features in the four available datasets. The hyphen symbol (-) is used in the table to indicate that the visual feature space is not applicable for the dataset.

6.3 Experiment 1: Learning Visual Concepts

In this section, we present empirical results to evaluate the visual concept extraction and learning framework. Visual concepts are abstractions of the feature spaces generated by the robot modalities which carry a human-level meaning such as a colour or a spatial relation. Visual concepts are learned automatically by clustering the low-level sensory input of each of the sensor modalities of the robot after an appropriate encoding. This clustering operation results in a collection of classes that are candidate visual concepts within each feature space. We incrementally learn concepts in each of the feature spaces; namely faces, colours, objects, locations, directions, distances, robot actions and human activities, over the four collected/extended datasets. Since the learning is performed in a loosely-supervised setting, and the robot does not know the label of each concept beforehand, then we use two popular clustering metrics to evaluate the performance: normalised *Mutual Information* (1991), and *V-measure* (2007). For the ground truth of each datasets, we use the sets presented in Table 6.2, extracted manually from each dataset by paid annotators.

As an upper bound and to provide a reference result, we also show the V-measure results obtained using a supervised (linear) support vector machine classifier (SVM) with 4-fold cross-validation. The SVM clearly has access to the ground truth labels during training. Still, in the following sections we show how the SVM only marginally outperforms our loosely-supervised visual concept learning framework in the four available datasets, even though our system learns visual concepts from unlabelled data.

6.3.1 Learning visual concepts results

Extended Train-Robots dataset

Table 6.3 presents results of our incremental, loosely-supervised visual concept extraction when compared against ground truth classes for the Extended Train-Robots dataset. This dataset contains 5 visual feature spaces, namely colours, shapes, locations, directions and robot actions. Using our visual concept extraction framework, the robot managed to recover 9 colour concepts, 5 shape concepts, 8 location concepts, 13 direction concepts and 5 robot action concepts from this relatively simple and simulated dataset. The numbers of manually defined unique concepts are presented in Table 6.2. The robot had access to noisy observation of the world, we added Gaussian noise to typical mean values of the observations. The number of unique concepts is selected unsupervised using BIC for simple visual concepts (colour, shape, location and direction), and using graph matching for complex concepts (robot actions). The results in Table 6.3 show the majority of the instances observed are successfully clustered into consistent concepts. Also, our system achieves comparable results to the supervised SVM, keeping in mind that we learn from unlabelled data. Examples of learned visual concepts are presented in Figure 6.14.

Metric	Colours	Shapes	Locations	Directions	Actions
Mutual Information	1.31	1.07	1.58	1.40	0.73
Normalised MI	0.70	0.70	0.54	0.58	0.72
Homogeneity Score	0.63	0.69	0.38	0.48	1.00
Completeness Score	0.77	0.70	0.76	0.71	0.54
V-measure	0.69	0.70	0.51	0.57	0.69
V-measure (SVM)	**0.77**	**0.79**	**0.57**	**0.91**	**0.89**

Table 6.3: Experimental results of visual concept extraction for the Extended Train Robots dataset, showing five clustering metrics. Also, we show the V-measure of a supervised SVM as an upper limit, that has access to the ground truth labels during training.

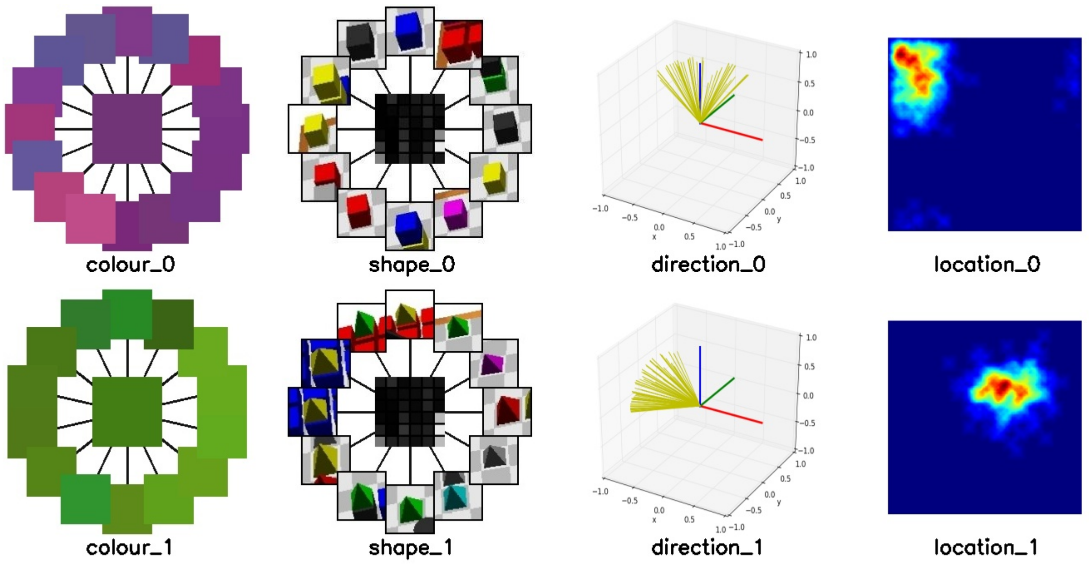


Figure 6.14: Examples of visual concepts learned from the extended Train Robots dataset. The examples include clusters of colours, shapes, directions and locations.

Leeds Robotic Commands dataset

Table 6.4 presents the results of learning visual concepts from the Leeds Robotic Commands dataset. This dataset contains 6 visual features, namely colours, shapes, locations, directions, distances and robot actions. Our system managed to recover 16 colour concepts, 25 shape concepts, 6 location concepts, 6 direction concepts, 4 distance concepts and 7 robot action concepts from this real-world dataset. The results in Table 6.4 show that the observed instances are reasonably clustered into consistent concepts. The number of learned concepts was selected unsupervised using BIC and graph matching approaches. For example, our robot thinks there are 25 unique shape concepts in this dataset, when in fact there are only 13 classes, and 7 robot action concepts when there are only 3. We found a number of reasons behind the larger number of recovered concepts when compared to ground truth data. First, using unsupervised object segmentation techniques (as presented in §Object detection 3.2.2) to identify the individual objects in the scene does not produce perfect object segments, which lead to having objects with incorrect point cloud segments (with extra or missing points/parts). Second, using a particle filter to track objects (as presented in §Object tracking 3.2.2) produced noisy tracks that lead to variations in activities. Third, objects were recorded from different view points which led to variations in their appearance. Objects were placed in different orientations on the table in each scene and were viewed from different angles from the camera. Fourth, objects were allowed to be partially occluded by other objects in the scenes. Fifth, the recordings of videos occurred at different times of the day with varying lighting conditions in the robotics lab which lead to variations in object colours. Finally, the same action was performed differently by different annotators, e.g. a simple pick up action was performed in various ways as annotators approached the objects from different angles, which lead to variations in the spatio-temporal graph structure. These reasons made learning of visual concepts from real-word data more challenging for our robot, yet, our system still managed to learn and cluster the visual concepts with comparable accuracy with the supervised SVM system, and it even produced better results in the direction relation feature space. Examples from the learned visual concepts for this dataset are shown in Figure 6.15.

Metric	Colours	Shapes	Locations	Distances	Directions	Actions
Mutual Information	1.46	1.19	1.27	1.13	1.16	0.82
Normalised MI	0.62	0.58	0.83	0.88	0.93	0.69
Homogeneity Score	0.68	0.62	0.81	0.91	0.91	1.00
Completeness Score	0.58	0.53	0.86	0.86	0.94	0.48
V-measure	0.62	0.57	0.83	0.88	**0.93**	0.65
V-measure (SVM)	**0.82**	**0.62**	**0.96**	**0.89**	0.90	**0.71**

Table 6.4: Experimental results of visual concept extraction for the Leeds Robotic Commands dataset, showing five clustering metrics for colour, shape, location, direction, distance and robot action extraction.

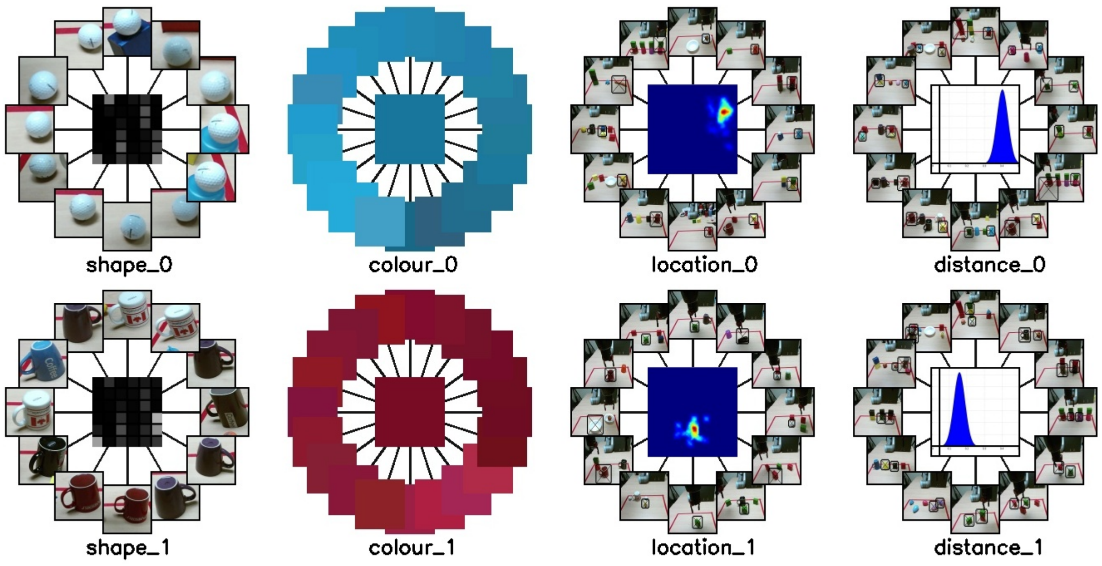


Figure 6.15: Examples of visual concepts learned from the Leeds Robotic Commands dataset, including clusters of shapes, colours, locations and distances.

Extended Object Ordering dataset

Table 6.5 presents the results of concept extraction for the Extended Object Ordering dataset. This dataset contains 3 visual features only, namely colours, shapes and robot actions. Note that more feature spaces can be added to this dataset such as weights and sizes, but we leave this as an extension to future work. Using our learning system, the robot managed to recover 7 colour concepts, 10 shape concepts and 7 robot action concepts from this relatively simple real-world dataset. The results in Table 6.5 show that the instances observed are successfully clustered into consistent concepts with good accuracy, even when compared to the supervised

SVM system. The action concepts results are relatively high for this dataset. The two main reasons for this high score are firstly the 7 actions in this dataset are simple, namely *grasp, lift, lower, drop, press, push* and *hold* with distinctive differences between them, and secondly these actions were programmed and executed by the robot when recorded, as opposed to being performed by an annotator guiding the robot arm which produces more variations in each action class. Examples from the learned visual concepts are shown in Figure 6.16.

Metric	Colours	Shapes	Actions
Mutual Information	1.19	0.77	0.78
Normalised MI	0.67	0.45	0.96
Homogeneity Score	0.66	0.57	0.96
Completeness Score	0.67	0.36	0.96
V-measure	0.67	0.44	**0.96**
V-measure (SVM)	**0.88**	**0.47**	0.93

Table 6.5: Experimental results of visual concept extraction for the Extended Object Ordering dataset, showing five clustering metrics for colours, shapes and robot actions extraction.

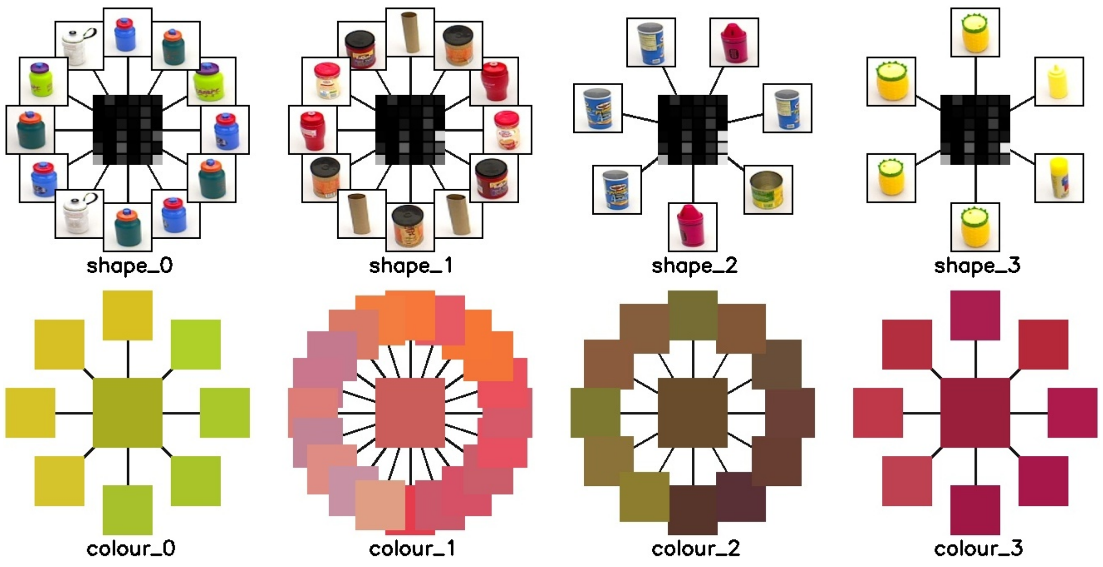


Figure 6.16: Examples of visual concepts learned from the extended Object Ordering dataset. The examples include clusters of shapes, colours and locations.

It is worth noting that the previous three datasets data collection, extension, processing and analysis were performed using a midrange PC. The robots were simulated/controlled using a single PC with an Intel Core i7-4790 processor and 16 GB of RAM running Ubuntu 14.04

OS and ROS indigo. The use of incremental Gaussian mixture model (IGMM) technique for simple concept learning allowed for smaller memory requirements, making it possible for the full incremental learning framework of visual concepts to run on-board a single PC.

Extended Kitchen Activities dataset

Table 6.6 presents results of our incremental, loosely-supervised visual concepts extraction when compared against ground truth classes for the Extended Kitchen Activities dataset. We use the most likely component in a mixture as a label if the prediction is multinomial, as in the case of activity topics. The robot managed to recover 34 face concepts, 13 colour concepts, 14 object concepts, and 13 activity concepts from this challenging real-world dataset with multiple view points, changing lighting conditions and occlusions. The results in Table 6.6 show the majority of the instances observed are successfully clustered into consistent concepts. Examples from the learned visual concepts for this dataset are shown in Figure 6.17.

Metric	Faces	Colours	Objects	Activities
Mutual Information	1.85	1.27	1.21	1.34
Normalised MI	0.70	0.70	0.69	0.62
Homogeneity Score	0.90	0.91	0.71	0.60
Completeness Score	0.55	0.54	0.68	0.64
V-measure	0.68	0.66	0.69	0.62
V-measure (SVM)	**0.75**	**0.74**	**0.77**	**0.69**

Table 6.6: Experimental results of unsupervised concept extraction showing five clustering metrics for face, colour, object and activity extraction. Also, we show the V-measure using a supervised SVM as an upper limit.

Given the limited size of the dataset, we compute the most prominent 20 Eigenfaces from the observations of day 1, and use them after that to compute Eigenvalues in all later detections. Also, we first seed the activity model by learning topics using Collapsed Gibbs Sampling (2014) on day 1 observations in batch mode. After that, we incrementally process new data using Variational Bayes with a regular mini-batch size of 5 videos to allow frequent updating. For the number of topics/human-activity concepts, we first start with the number of discovered objects to initialise the learning, then increase this number by one each day to allow new activities to appear over time. Also, we remove any unused topics.

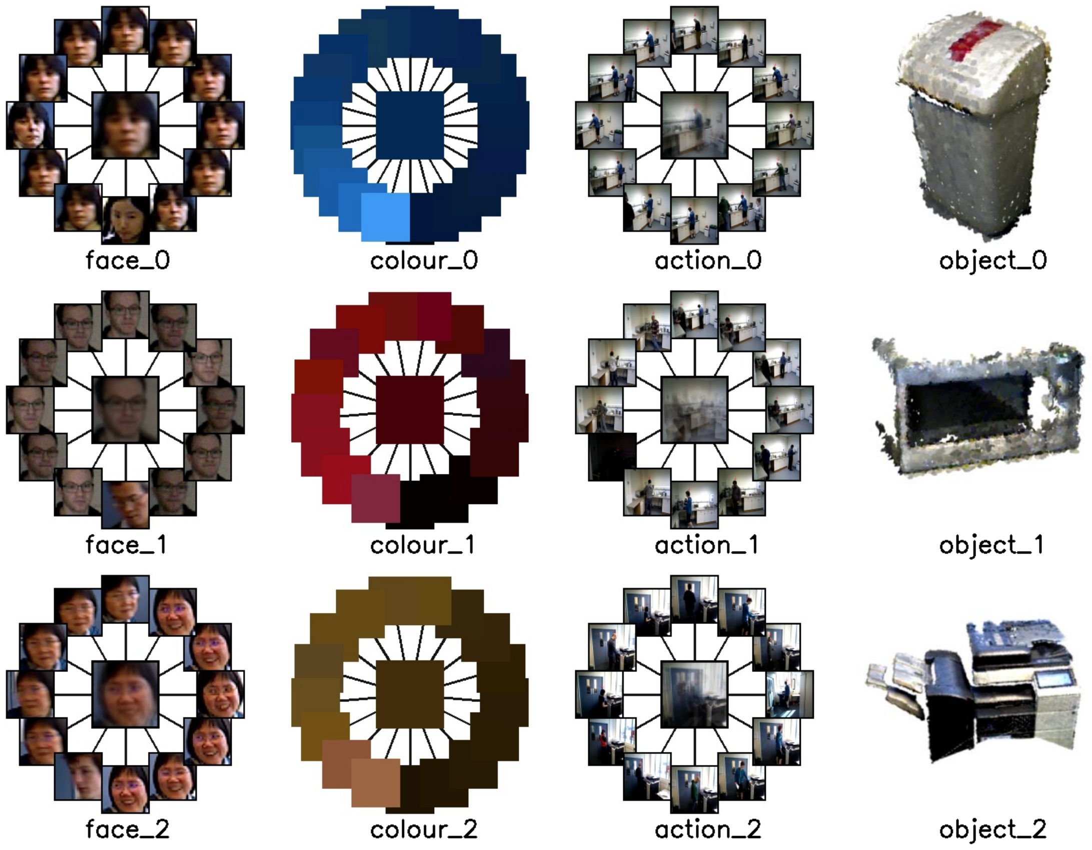


Figure 6.17: Examples of visual concepts learned from the Extended Kitchen Activities dataset. The examples include clusters of faces, colours, human activities and segmented objects.

It is worth noting that all data collection, processing and analysis for this dataset were performed using midrange CPU and GPU units. Our mobile robot LUCIE has three PCs with i7 processors running ROS indigo, and a single GTX 1050 Ti GPU with 2 GB of memory on which the convolutional pose machine (CPM) for human pose estimation runs. The use of incremental techniques (IGMM and VB) for concept learning allowed relatively less complex and more memory-efficient processing, making it possible for the full framework to run on-board the PCs of our robot.

6.3.2 Discussion

The visual concepts learning results obtained from all four datasets show that our system is capable of learning visual concepts from robot observation in a loosely-supervised manner. The term loosely-supervised is used to describe the fact that video clips were temporally segmented by human annotators such that each clip contains a single human or robot action. These temporally segmented clips were used for both crowd-sourcing of textual descriptions and learning. Our system succeeded in extracting and learning meaningful concepts even from real-world challenging datasets were objects and people were viewed from different angles and with occlusions. The results obtained are also comparable with the ones generated using a supervised SVM approach, keeping in mind that the SVM had access to the human annotated ground-truth labels during learning while our system learns from unlabelled data. The visual concept learning results generated from our system can be improved using semi-supervised learning approaches, where the robot is provided with a few data points with labels and many other data points without labels. The labels can be obtained/extracted from video clips that contain a single object in them, or even using human-robot interaction where the robot asks about a specific feature space such as the colour of a few objects to improve the learning. We leave the aforementioned extensions for future work to investigate. The learned visual concepts presented in this experiment are used in the following section to learn language groundings.

6.4 Experiment 2: Learning Language Groundings

In this section, we present empirical results to evaluate the natural language grounding framework presented in this thesis. Natural language grounding is a term used to refer to the problem of learning the meaning of words in other domains by mapping words to concepts in these domains. Our language grounding framework is presented in Chapter 4 and consists of a preprocessing step and four learning steps as shown in Figure 6.18. In the preprocessing step, we extract meaningful concepts from both vision and language inputs. The vision concepts are represented with Gaussian components, human activity topics and spatio-temporal DAGs as discussed in Chapter 3, while the language concepts are represented with a bag of n-grams that covers all word sequences up to length N in the input sentences. The aim of our language grounding framework is to map the extracted n-grams to their corresponding visual concepts, which is achieved using the four learning steps: (i) building associations, (ii) generating hypotheses, (iii) filtering hypotheses, and (iv) validating hypotheses.

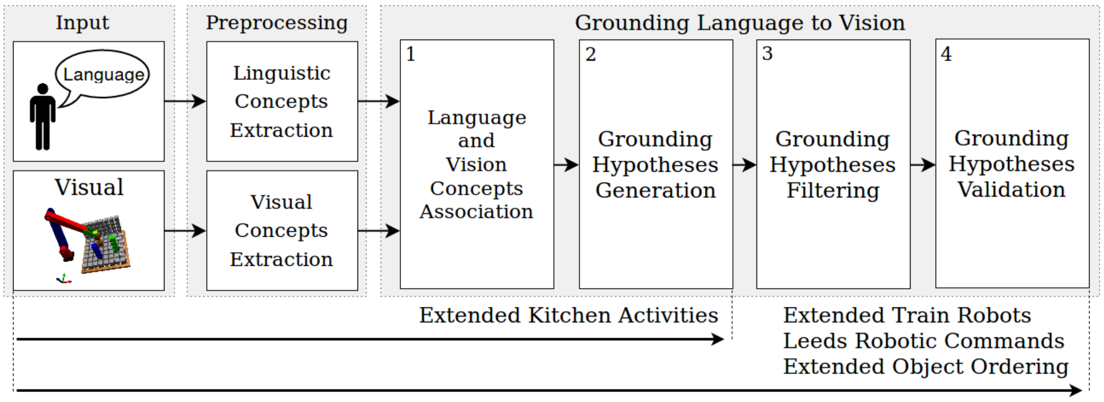

Input
Language
Visual
Preprocessing
Linguistic Concepts Extraction
Visual Concepts Extraction
Grounding Language to Vision
1
Language and Vision Concepts Association
2
Grounding Hypotheses Generation
3
Grounding Hypotheses Filtering
4
Grounding Hypotheses Validation
Extended Kitchen Activities
Extended Train Robots
Leeds Robotic Commands
Extended Object Ordering

Figure 6.18: Natural language grounding framework. The framework consists of a preprocessing step, and four learning steps aiming to find the correct mapping between n-grams in language and visual concepts in vision. The arrows at the bottom shows the different processing steps applicable to each of the four datasets.

In our language grounding framework, we allow multi-to-multi mappings between language and vision to preserve the richness of language. In other words, a vision concept can be grounded with different n-grams and visa versa. For example, the n-grams '*cyan*' and '*sky blue*' can share the same colour concept. Similarly, the n-gram '*Tony*' can be shared with multiple face concepts (i.e. different people can be called 'Tony' and they may look different). The multi-to-multi mappings are enabled by formulating the grounding problem into an integer programming one as described in §Grounding Hypotheses Generation (4.4), which is one the key novelties of this work. Another key novelty of this work is the validation process of grounding hypotheses. The validation is achieved using a mental simulation technique developed to find the correct groundings of language to vision as described in §Grounding Hypotheses Validation (4.6). Mental simulation is a term first articulated by Craik (1967) and is used to describe how humans evaluate their environment to better understand the interactions between its components. Our validation approach is motivated by Craik's proposal, that the brain builds mental models that support inference by mental simulations, to help the robot verify which of the available grounding hypotheses are correct and which are not. The validation process is enabled by using graph matching techniques to validate the grounding hypotheses.

The grounding results for the first three datasets (Extended Train-Robots, Leeds Robotic Commands and Extended Object Ordering) are obtained using the full framework and computed using n-grams of length less than or equal three, $n \leq N, N = 3$. On the other hand, the results for the fourth dataset (Extended Kitchen Activities) are obtained using only the first two steps in the framework, and computed using only words (i.e. n-grams of length one, $N = 1$) as shown in Figure 6.18. The fourth dataset is treated differently due to using a different approach to learn about human activities, i.e. using LDA and Variational Bayes instead of spatio-temporal DAGs to model human activities. In Chapter 3, we discussed how learning human activities differs from learning robot actions as it requires more elaborate encoding and more sophisticated clustering mechanism to model the variation in each activity class, and how LDA and Variational Base algorithms are used to model these activity classes. Moreover in Chapter 4, we discussed how using LDA prevents the robot from executing/repeating the learned actions, as the specific ordering of how to perform the action is lost. As a result, the ability of validating the grounding hypotheses is lost, the system is limited to the first two steps of the grounding framework.

6.4.1 Natural language grounding results

We present the empirical results for our language grounding framework demonstrating its ability to acquire correct groundings from pairs of short video clips and their corresponding descriptions. We aim to learn all the possible groundings of words to their corresponding visual concepts. For ground truth, we manually annotated all correct word-vision groundings for each of the learned visual concepts in the four listed datasets, e.g. the word '*red*' should be grounded to the learned Gaussian component of the colour red, and the phrase '*pick up*' should be grounded to the learned spatio-temporal DAG of the pick up action, etc. The learning begins by feeding the recorded video clips and sentences to our system incrementally, effectively updating the robot's knowledge in language grounding. As a metric, we compute the F1-score (1979) of the grounding results in each feature space separately. The F1-score penalises both incorrect and missing groundings between language and vision, therefore providing a better insight than precision or recall into our grounding framework.

As an upper bound, we also present the results obtained using a supervised Hidden Markov Model (HMM) for Part-of-Speech (POS) tagging system presented by Rabiner (1989) with a Python implementation available in the NLTK repository (2017). The HMM technique is desirable for POS tagging tasks as the highest probability tag sequence can be calculated for a given sequence of words. The use of an HMM differs from other POS tagging mechanisms which often tag each word individually without regard to the optimal combination of tags for the whole sentence. The HMM produce a sequence of tags for the input sentence using the Viterbi algorithm, which efficiently computes the optimal path through the graph given the sequence of input words. The HMM has access to ground truth data/tags during training. The HMM requires for learning both the input sentences (e.g. "*move the red sphere over the green block*") and the annotated tags (e.g. "*action, none, colour, shape, none, colour, shape*"). A four fold cross validation is performed to compute the F1-scores for the HMM system on all four datasets.

Table 6.7 presents the final F1-scores computed using our incremental learning framework and the supervised HMM system for each of the four datasets. The results show how our system was able to successfully learn part of the correct language groundings in each dataset. It also shows that our system achieves comparable results with the supervised HMM system even though it learns from unlabelled sentences.

Natural language grounding results (F1 scores)

Datasets	Train Robots		Leeds Commands		Object Ordering		Kitchen Activities	
Systems	OS	HMM	OS	HMM	OS	HMM	OS	HMM
Colour	0.92	0.93	0.42	0.84	0.81	0.92	0.57	0.84
People	-	-	-	-	-	-	0.47	0.68
Object	0.88	0.94	0.51	0.81	0.86	0.95	0.43	0.76
Location	0.62	0.80	0.34	0.78	-	-	-	-
Direction	0.55	0.89	0.56	0.82	-	-	-	-
Distance	-	-	0.55	0.93	-	-	-	-
Action	0.82	0.91	0.59	0.91	0.78	0.96	0.55	0.90

Table 6.7: Natural language grounding results. OS stands for Our System and HMM stands for the Hidden Markov Model system. The hyphen symbol (-) is used in the table to indicate that the visual feature space is not applicable for the dataset.

Figure 6.19 shows the language grounding incremental results obtained using our system from each of the four datasets. The graphs show an improving trend in the F1-score of the word groundings in each feature space as more data is observed/processed. We hypothesise that extended observation of the environment will allow all the concepts in these predefined feature spaces to be correctly grounded in a loosely-supervised manner. Similarly, the visual concepts themselves will improve with more observations.

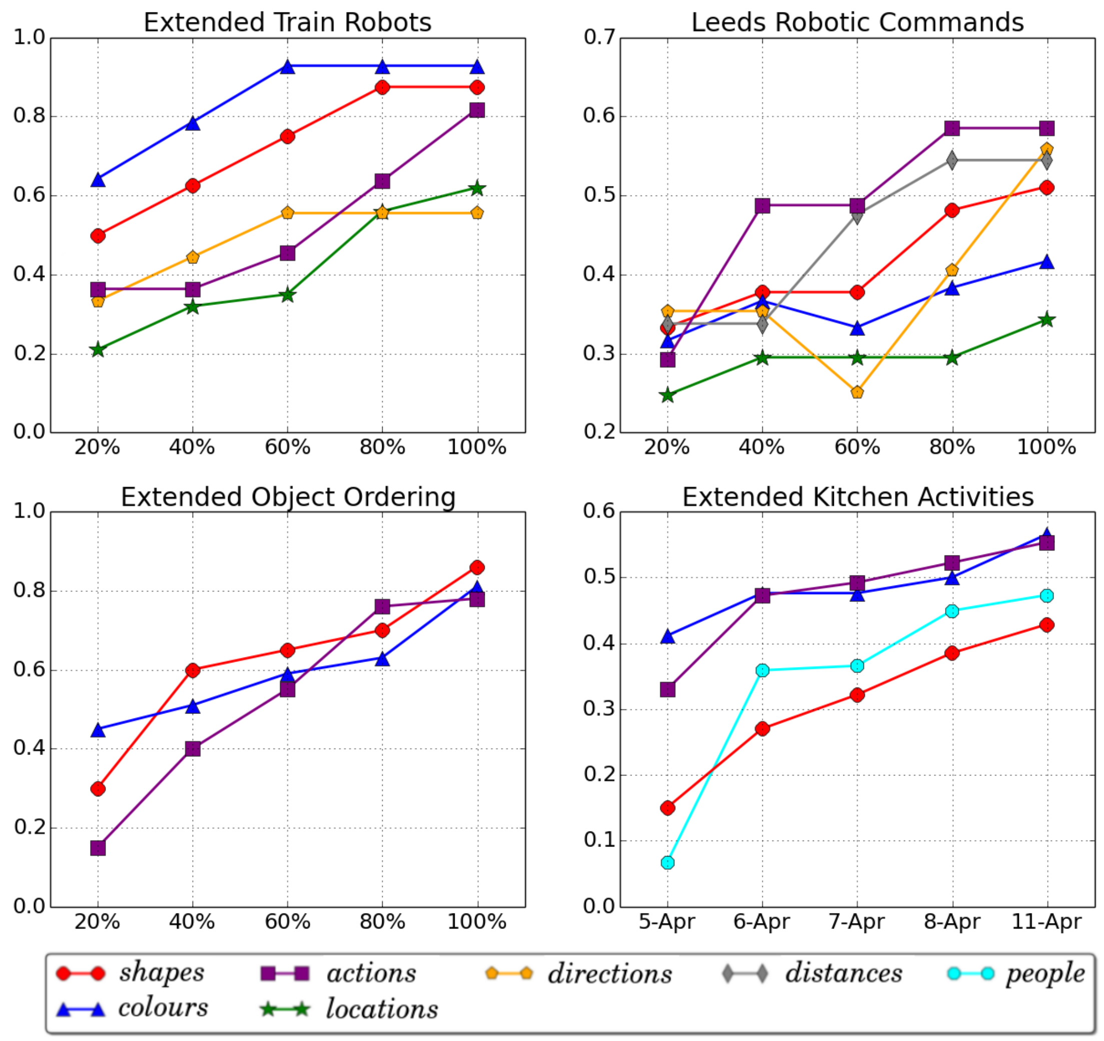


Figure 6.19: F1-scores for incremental language grounding for each dataset. The Extended human activity dataset collected using LUCIE is processed using daily batches, i.e. we increment the learning by processing the videos collected in each day. Note that different y-axes scales were selected for each dataset to better show the results.

Figure 6.20 shows examples from the learned language and vision groundings using our incremental system for each of the four datasets. The examples show how each robot managed to learn the mappings between language and vision. The system did not learn the groundings of all n-grams in the datasets due to noise or lack of training data. For example, the 1-gram *cyan* in the Leeds Robotic Commands is mentioned only once in the entire dataset and therefore our system did not manage to ground it with its corresponding colour concept. Similarly, our system learned some incorrect mappings, such as mapping the 1-gram "*glasses*" to a person wearing glasses thinking that *glasses* is the name of that person. We hypothesise that extended observation of the environment will allow all the groundings in these predefined feature spaces to be correctly learned in a loosely-supervised manner without the need for human annotations.

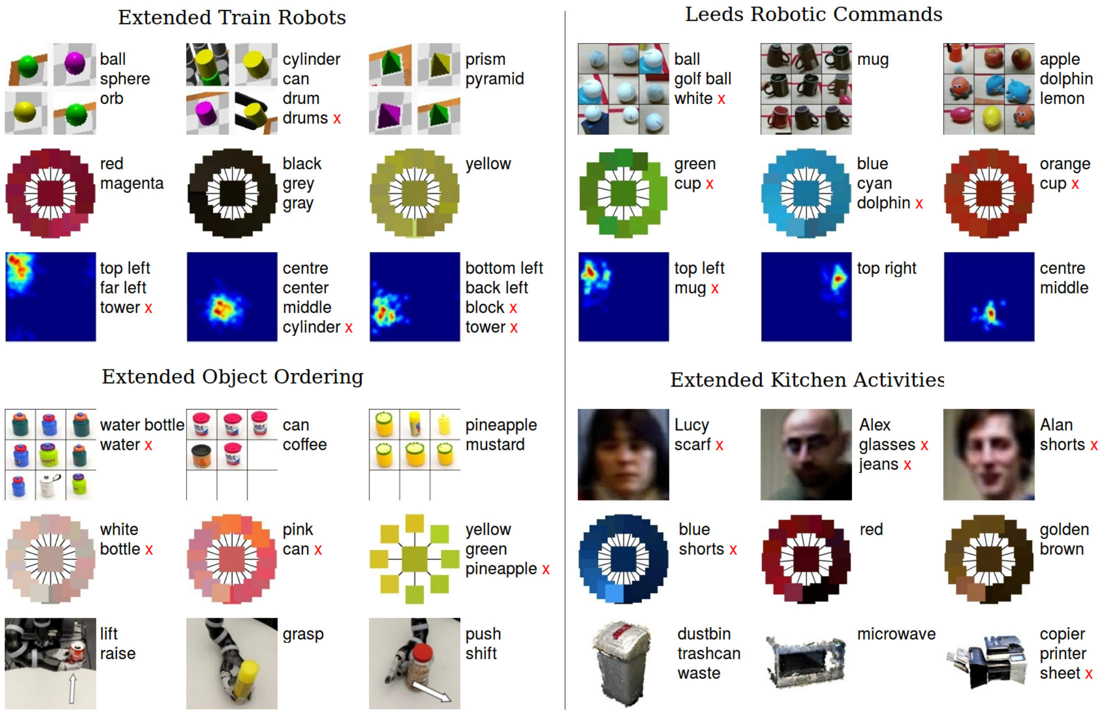


Figure 6.20: Examples of learned language groundings from all four datasets. Note that the cross mark symbol (✗) is used to indicate that this learned grounding is incorrect. We manually annotated the list of correct groundings for each visual concept.

6.4.2 Grounding in other languages

In this section we evaluate our language grounding framework in learning from other languages. We translated all 4485 commands in the Extended Train Robots dataset from English to Arabic. The translation was performed using Goslate (2016), a free Google translator API for the Python programming language. Particular care was taken not to alter the commands or correct any mistakes in them before translation. Examples of the translated commands were shown earlier in Figure 6.3. The learning framework is applied on the Arabic language in the exact same way as the English language. The translated commands are processed to extract all available n-grams from them, with $n \leq N, N = 3$. These n-grams are then used to learn the language grounding with vision as described in Chapter 4. Table 6.8 presents the results of language grounding in both Arabic and English for the Extended Train Robots Dataset. As an upper bound, we present the results obtained using a supervised Hidden Markov Model (HMM) for POS tagging. The HMM system has access to the ground truth data and was tested in a four-fold cross validation setup.

Language grounding in Arabic

Language	Arabic		English	
Systems	OS	HMM	OS	HMM
Colour	0.78	0.89	0.92	0.93
Object	0.84	0.90	0.88	0.94
Location	0.55	0.70	0.62	0.80
Direction	0.54	0.81	0.55	0.89
Action	0.80	0.88	0.82	0.91

Table 6.8: Natural language grounding results for Arabic language in the Extended Train Robots dataset. OS stands for Our System and HMM stands for the supervised Hidden Markov Model system. The English grounding results are presented for comparison with the Arabic ones.

The results in Table 6.8 show that our system performed well in comparison with the supervised HMM system in learning from the Arabic language. The F1-scores are slightly worse in learning from Arabic than in English. We believe the reason behind this is that nouns, verbs, and adjectives have genders in the Arabic language. The grammatical gender in Arabic is one of two: a word may be masculine or feminine, and there is no neuter option. Moreover, masculinity is the default grammatical gender and a word does not have to have anything special in order to reflect this (e.g. masculine grey $\rightarrow$ ramady). Femininity on the other hand, is not default and a word would have to have something special added to it to reflect this gender (e.g. feminine grey $\rightarrow$ ramadia, the *ia* added to the end of this word to reflect the femininity). Therefore, the same word appears in multiple forms in the dataset and are treated differently in our system. The masculine grey word and the feminine grey one are considered two different n-grams, and are mapped to visual concepts separately. This results in having fewer training examples for the same word when compared with the English language. With this in mind, our system still managed to ground words in Arabic to their corresponding visual concepts. Examples of learned groundings from both the Arabic and English language are shown in Figure 6.21. The arrows are used to indicate the direct translation between the two words. This means that our system can be used to learn translation between languages based on the their groundings to the visual domain, but we leave this idea open for future work to investigate and validate.

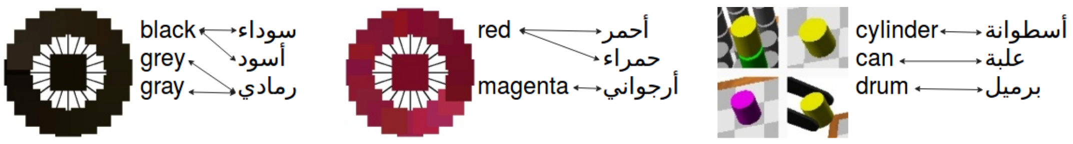
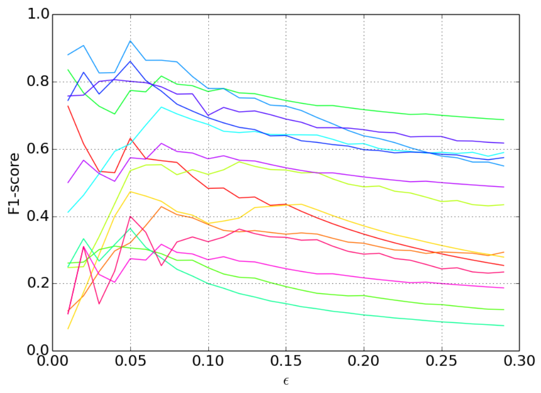


Figure 6.21: Examples of learned language groundings from both Arabic and English language in the Extended Train Robots dataset. The training was performed on each language separately. The arrows between words are used to indicate the direct translation between the two words and were manually added to the image. Our system does not know that these words are translations in different languages.

6.4.3 Sensitivity analysis for grounding the parameter (ϵ)

In this section we discuss the sensitivity analysis performed for the language grounding parameter epsilon (ϵ). We formulate the natural language grounding problem into an integer program, which has the benefit of allowing multi-to-multi mappings between words in language and concepts in vision. The parameter epsilon presented in §Hypotheses Generation (4.4) is used as a threshold to keep the sparsity of groundings by forcing the number of selected groundings to be below this threshold. All language grounding results presented in this work were obtained using an $\epsilon = 0.05$. The selection of this value was based on a sensitivity analysis experiment performed over all four datasets. Figure 6.22 presents the results of this experiment. The results of this experiment shows that the grounding performance peaks at $\epsilon = 0.05$ for most of the feature spaces in the four datasets, and therefore this value was selected in this work.



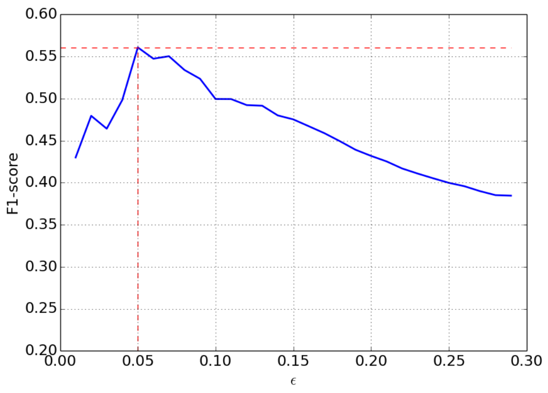


Figure 6.22: Sensitivity analysis for the language grounding parameter epsilon (ϵ). (left) the graph shows the final F1-score values in each feature space from the four datasets on the y-axis, and the different ϵ values used to compute these F1-scores on the x-axis. (right) the average F1-score results obtained from all feature spaces.

6.5 Experiment 3: Learning Grammar Rules

We evaluate our grammar induction framework based on its ability to learn grammar rules capable of parsing never-seen-before linguistic commands. Grammar induction refers to the process of learning a formal grammar, usually as a collection of re-write rules or productions from a set of observations. The observations usually consist of natural language sentences annotated with grammar trees. The human/manual annotation of grammar trees is a labour intensive task that hinders learning from large datasets. In this work, we do not use human annotations to learn the grammar rules, but rather we employ the visual inputs (video clips) to infer the grammar structure of the input sentences as presented in Chapter 5.

We use Robot Control Language (RCL) trees to represent the learned structure of input sentences. RCL is a tree semantic representation for natural language commands. Each sentence is represented as an RCL tree, where leaf nodes align to words in the corresponding sentence, and non-leaves are labelled with a predefined set of categories that the robot can understand and execute. The predefined RCL labels used in this work are the visual features defined in Chapter 3 (e.g colour, shape, direction, etc.) and are limited to robot manipulation tasks.

The aim of our system is to automatically generate an RCL tree from each input video-sentence pair. Each input video is used to extract a vision tree (Ω), which is used to infer the structure of the input sentence and generate a grammar tree. We automatically generate a grammar tree (an RCL tree) using the same idea used in validating the language groundings, which is comparing the language model with the vision input. This idea assumes that the input sentences provided to the robot are describing the actions, objects and relations involved in the corresponding input video clip. Therefore, the sentence structure should also reflect/map the features extracted from the input video. The generation of grammar trees from input sentences and videos is formulated as a search problem presented in Algorithm 1, which is the key contribution offered by this work in the field of grammar induction. The algorithm takes as inputs the extracted vision tree (Ω) and the input sentence, and outputs the grammar tree (Ψ).

The resultant grammar trees ($\Psi_1, \Psi_2, \ldots$) are then used to learn the grammar rules of natural language in the form of grammar $\Pi = (N, T, R, S, P)$, which provides our robot with the ability of understanding new linguistic commands by parsing them into RCL trees.

We use Probabilistic Context Free Grammar (PCFG) to model the grammar rules. We start with an empty rule set, and incrementally create and update the rules observed in each generated RCL tree (Ψ). The creating/updating of grammar rules is achieved using the Inside-Outside algorithm presented by Lari and Young (1990) as described in §Learning Grammar Rules (5.4). The rules from all grammar trees are incrementally accumulated into a grammar set Π.

6.5.1 Grammar induction results

To evaluate our grammar induction framework and the learned grammar (Π), we test on three robotic manipulation datasets, (i) Extended Train Robots, (ii) Leeds Robotic Commands, and (iii) Extended Object Ordering datasets. Each of the three datasets is randomly divided into four folds, and four fold cross-validation is applied. Three folds are used to learn grammar rules, and the fourth is used for testing, we repeat this process four times to test on each fold. The learned grammar rules are evaluated based on their ability to correctly parse new (never seen before) linguistic commands. A parser is equipped with the learned grammar set (Π) and is used to parse the commands in the test fold.

The results present the score of correctly parsed RCL sub-trees from sentences in each of the test fold. A score of 1 is given if the parsed sentence completely matches the human annotation, while a partial score in (0, 1) is given if it partially matches the human annotation.

The partial matching is computed by matching subtrees in both trees divided by the total number of subtrees. For example, if a parsed tree contains 10 subtrees and only 8 of which match in links and labels with the manually annotated tree, then it is given a score of 0.8.

As an upper bound, we also present the results obtained using a supervised grammar induction system presented by Abney (1996). The supervised system has access to the human annotated RCL trees to learn the grammar rules from, while our system automatically generates them using Algorithm 1. Our system only uses the human annotation in the evaluation process. The same four fold cross validation procedure is applied on this system.

We also tested our system against an unsupervised grammar induction approach presented by Ponvert *et al.* (2011) that learns from language alone. This approach learns a language model via chunking the raw text into smaller parts that shows a repeated pattern throughout the dataset. Both our system and Ponvert's learn from unlabelled sentences, i.e. without the human annotated RCL trees. However, we learn from language and vision inputs, while Ponvert's system learns from language alone. We evaluate Ponvert's unsupervised system based on its ability to chunk the text into correct sub-trees only as it does not generate labels.

The grammar induction results for the three systems (i) Abney's supervised, (ii) ours, and (iii) Ponvert's unsupervised systems are presented in Table 6.9, for the three manipulation datasets. The results in Table 6.9 clearly show that our approach outperforms the unsupervised grammar induction system and achieves comparable results to the supervised system by learning from both language and vision as opposed to learning from language alone. The number of grammar rules generated differs between techniques as shown in the last row in Table 6.9. The supervised rules are higher in number because a few sentences contain classes which we assume we can not learn (the system fails to generate a grammar tree from the input sentence). For example, in the Extended Train Robots dataset there exist an indicator class for superlatives, e.g. (indicator $\rightarrow^{w}$ tallest). These classes add to the number of learned rules. However, the results do not vary as much because there are not many sentences including these rules.

An example from one of the test commands in the Extended Train Robots dataset is presented in Figure 6.23. The example is for the command "*place the yellow ball on top of the blue cylinder*". The figure shows the parsed tree using the learned grammar set (Π) from our approach (top), and the parsed tree using Ponvert's unsupervised system that learns from lan-

Grammar induction results									
Datasets	Train Robots			Leeds Commands			Object Ordering		
Systems	US	OS	SS	US	OS	SS	US	OS	SS
Fold 1	0.32	0.66	0.74	0.28	0.83	0.88	0.41	0.86	0.91
Fold 2	0.31	0.65	0.74	0.28	0.81	0.89	0.42	0.81	0.89
Fold 3	0.35	0.66	0.75	0.30	0.83	0.87	0.41	0.85	0.91
Fold 4	0.31	0.69	0.75	0.28	0.84	0.87	0.40	0.84	0.90
average	0.32	0.66	0.75	0.28	0.83	0.88	0.42	0.84	0.90
grammar rules	45	46	114	38	39	89	30	34	78

Table 6.9: Grammar induction results. US stands for Unsupervised system (Ponvert's), OS for Our System, while SS stands for Supervised System (Abney's). The values presented in the table are percentage of correctly parsed subtrees in each test fold. The last raw presents the average number of grammar rules or productions generated in all four folds.

guage alone (bottom). The learned grammar rules from our system used to parse this natural language command are presented in Table 6.10. The rules are written in the form ($lhs \rightarrow^{w} rhs$), where lhs is the left had side of the grammar rule, rhs is the right hand side, and w is the weight of the rule. For example, the grammar rule (colour $\rightarrow^{0.16}$ *yellow*) is used to tag the word *yellow* as a colour non-terminal, similarly rule (shape $\rightarrow^{0.13}$ *ball*) is used to tag the word *ball* as a shape non-terminal, while the rule (entity $\rightarrow^{0.85}$ colour, shape) is used to group both non-terminals (colour and shape) as the non-terminal entity. The parser loops through all learned rules to maximise the final probability value of the parsed tree using the CYK algorithm. The CYK algorithm or the Cocke-Younger-Kasami algorithm (1967) is a parsing algorithm for context-free grammars that employs bottom-up parsing and dynamic programming. Note that our robot is not assumed to know the words *colour, shape, entity, etc.* specifically, but rather knows of the existence of these elements or types (since it already has feature spaces for each of these).

6.5.2 Grammar induction in other languages

In this section we evaluate our grammar induction framework in learning from other languages. We translated all 4485 commands in the Extended Train Robots dataset from English to Arabic as previously discussed in §Language grounding in other languages (6.4.2). Examples of the translated commands were shown earlier in Figure 6.3. The learning of grammar rules is

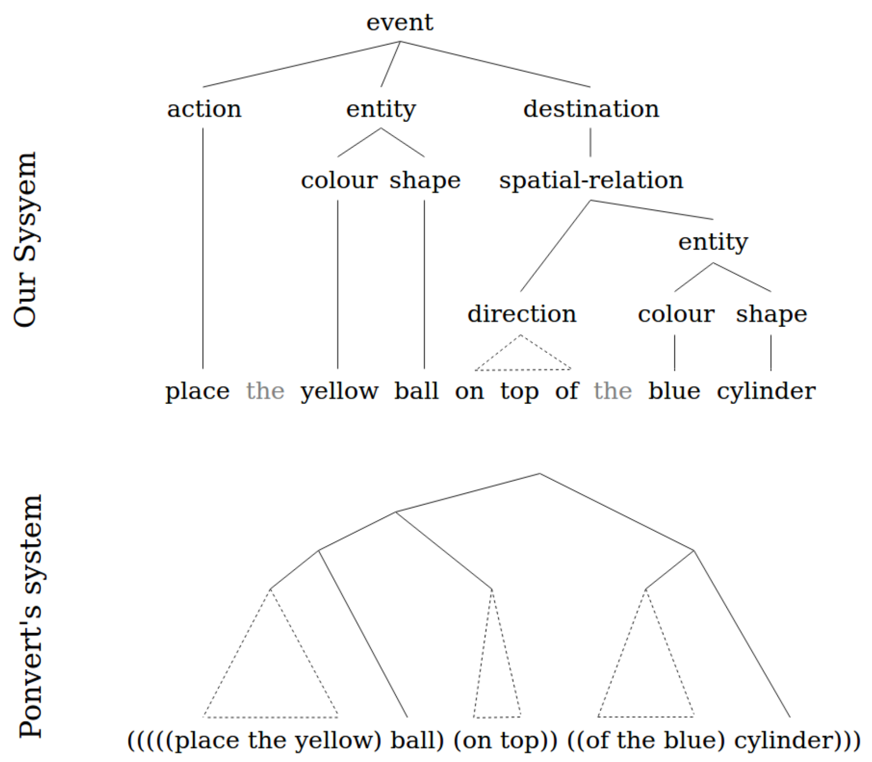

Our Sysyem
event
action
entity
destination
colour
shape
spatial-relation
entity
direction
colour
shape
place the yellow ball on top of the blue cylinder
Ponvert's system
(((((place the yellow) ball) (on top)) ((of the blue) cylinder)))

Figure 6.23: The grammar trees generated for the new command "*place the yellow ball on top of the blue cylinder*" using our system (top) and Ponvert's unsupervised system (bottom).

Terminal Leaves	*Non-Terminals*
colour $\rightarrow^{0.16}$ *yellow*	event $\rightarrow^{0.33}$ action, entity, destination
colour $\rightarrow^{0.22}$ *blue*	entity $\rightarrow^{0.16}$ colour, shape
shape $\rightarrow^{0.13}$ *ball*	destination $\rightarrow^{0.81}$ spatial-relation
shape $\rightarrow^{0.05}$ *cylinder*	spatial-relation $\rightarrow^{1.0}$ direction, entity
action $\rightarrow^{0.01}$ *place*	
direction $\rightarrow^{0.52}$ *on top of*	

Table 6.10: The learned grammar rules used to parse the command "*place the yellow ball on top of the blue cylinder*" shown in Figure 6.23 (top).

applied on the Arabic language in the exact same way as the English language. The translated commands are processed to extract all available n-grams. These n-grams are then used to learn the language grounding with vision as described in Chapter 4. Then, these language groundings are used to learn the grammar rules as presented in Algorithm 1. Table 6.11 presents the results of grammar induction in both Arabic and English for the Extended Train Robots Dataset. As an upper bound, we present the results obtained using a supervised system that has access to the human annotated RCL trees during training. We also compare our results with Ponvert's unsupervised system trained on the Arabic sentences. Even though it was not designed to work with the Arabic language, Ponvert's system was tested on other languages such as German and Chinese as presented in his paper (2011). The results in Table 6.9 show that our approach outperforms the unsupervised grammar induction system by learning from language and vision data. We also achieve results that are little lower than those of the supervised system but still very promising by learning from unlabelled data (without the human annotated RCL trees), as opposed to learning from labelled linguistic inputs. Moreover, this experiment shows that our system is capable of learning grammar rules regardless of the POS tags ordering in a sentence. For example, in the Arabic language, the adjectives comes after the noun in a sentence, while in English they are positioned before nouns. Our system succeeded in learning grammar rules that represent nouns and adjectives in both Arabic and English.

Grammar induction results in other languages

Language	Arabic			English		
Systems	US	OS	SS	US	OS	SS
Fold 1	0.31	0.62	0.69	0.32	0.66	0.74
Fold 2	0.30	0.62	0.70	0.31	0.65	0.74
Fold 3	0.32	0.59	0.70	0.35	0.66	0.75
Fold 4	0.31	0.61	0.69	0.31	0.69	0.75
average	0.31	0.61	0.70	0.32	0.66	0.75
grammar rules	57	63	187	45	46	114

Table 6.11: Grammar induction results. US stands for Unsupervised system (Ponvert's), OS for Our System, while SS stands for Supervised System (Abney's). The values presented in the table are the percentage of the correctly parsed sentences in each fold. The last row presents the average number of grammar rules generated in all four folds.

6.6 Experiment 4: Scalability and Memory Requirements

In this section, we present empirical results to evaluate the scalability of our language and vision learning framework presented in this thesis. Scalability refers to the capability of a system, network, or process to handle a growing amount of work, or its potential to be enlarged to accommodate that growth. Scalability is an important aspect in any life-long learning system, such as the system presented in this thesis for teaching robots about language and vision.

We evaluate the scalability of the three main components in our system, (i) visual concepts learning, (ii) natural language grounding, and (iii) grammar induction. The scalability of each component is evaluated using the memory requirement of the learned model compared with the size of the processed raw data. All calculation were measured on a Desktop PC, with an Intel Core i7-4790 processor with 8 cores and 3.6 GHz clock speed, and 16 GB of RAM.

We define the memory requirement of each component to be equal to the memory size of its learned model when stored on the PC's hard-drive. For example, the memory size of the Gaussian mixture models used to learn the colours, shapes, etc., or the memory size of the learned grammar rules, etc. Since our system learns incrementally, the learned models will grow in size with every video-sentence pair. Figure 6.24 shows the incremental memory requirement of the three components in our system along with the raw size of the input data in the Leeds Robotic Commands datasets. The graphs in Figure 6.24 show how efficient our learning system is when compared with the size of the raw data. The sizes of the learned models are orders of magnitude smaller than that of the raw data. For example, at the final video (video number 204) in the Leeds Activity Commands dataset the processed raw data was nearly a hundred Gigabytes in size, while the learned models did not exceed 50 Kilobytes in size. Moreover, the learned models memory requirements flatten as more data is observed, this is mainly because the system has learned most of the visual and linguistic concepts there are to learn in this dataset. Most of the vision concepts, words' groundings, and grammar rules have been observed and allocated a location in memory. We hypothesise that extended observation of the environment will scale well with our system as the size of the learned models will not increase linearly with the size of observed data, but rather would flatten as the robot explores most of the new concepts in its environment.

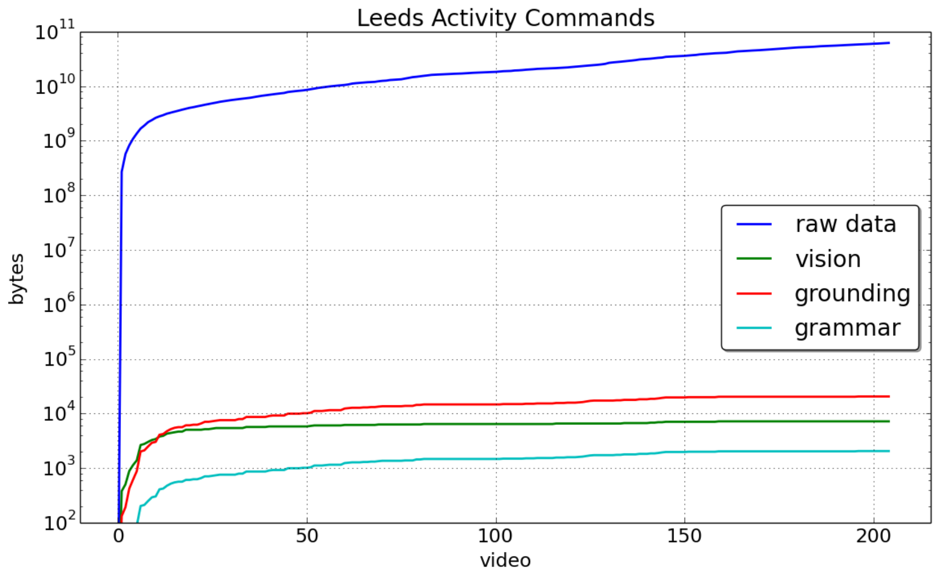


Figure 6.24: The incremental memory requirement of our learning system in the Leeds Robotic Commands dataset.

6.7 Summary

We have presented four experiments to evaluate our incremental loosely-supervised learning system. The experiments focused on evaluating four aspects of our system that include learning of (i) visual concepts from raw input videos, (ii) language grounding from raw video and text, (iii) grammar rules of natural language, and (iv) scalability analysis by measuring the memory requirements of our learned models. The results obtained from the four experiments show how our system is capable to bootstrap its knowledge in both language and vision in a loosely-supervised manner. We also compared our system against other supervised and unsupervised approaches to better demonstrate the limitations and abilities of our system. In the following chapter we discuss some of these limitations and suggest solutions in the form of future work research directions.

Chapter 7

Discussion and Conclusion

"Beautiful is what we see, more beautiful is what we know, most beautiful, by far, is what we don't."

—*Nicolas Steno*

We have presented a novel, incremental and loosely-supervised framework that enables robots to bootstrap their knowledge in language and vision domains. We have demonstrated for the first time in a developmentally plausible setting, that a system can concurrently and incrementally learn three kinds of knowledge:

– Visual representations of the world in a number of predefined feature spaces (objects properties, people attributes, spatial-relations, robot actions and human activities).

– Language grounding which maps words and phrases in language to their corresponding learned visual concepts.

– Probabilistic grammar rules of natural language.

Previous systems are designed to use one of these three components (visual representation, language grounding and language grammar) to learn the other two. To the best of our knowledge, this is the first system capable of learning all three components concurrently, which reduces the amount of predefined initial knowledge significantly. We also show that these components can be learned from real-world noisy data collected using mobile and manipulator robots equipped with different sensing modalities, thus enabling our robots to bootstrap their knowledge in both language and vision domains without the need for direct supervision.

7.1 Summary

Our learning framework is divided into three components: visual representation, language grounding and grammar induction. We assume little information is given initially, and learn the vision concepts by clustering the sensory-motor experience of each robot, and learn the language grounding and grammar by mapping natural language sentence to the vision domain. Out system is capable of "generalising" what it learned to unseen situations. For example, our probabilistic grammar rules are capable of parsing never-seen-before sentences by mapping words into their visual categories. For example, the learned rule that parses a 'colour' followed by 'shape' into an 'entity' (entity $\rightarrow^{w}$ colour, shape) can be applied to any colour and shape words, even if not all combinations are seen in the dataset, such as a "purple banana", or a "blue apple", etc. In the following sections, we summarise each of our learning components and highlight the key contributions in each.

7.1.1 Visual concepts

The learning of visual concepts is the first step in our language and vision learning framework. Visual concepts are learned automatically by clustering the low-level input of each of the robot's sensing modalities after an appropriate encoding. This clustering operation results in a collection of classes that are candidate visual concepts in each feature space. Because we assume no prior knowledge of the structure of the sensor feature spaces, we employ probabilistic modelling techniques to each feature space independently to elicit meaningful classes that are supported by the observed data.

We differentiate between two kinds of visual concepts. Simple concepts are ones that are time-independent and can be detected from a single or a small number of observations. For example, simple visual concepts like colours can be represented as Gaussian components in a Gaussian Mixture Model over the HSL space. On the other hand, complex concepts exhibit a temporal dimension and manifest over longer sequences of observations. For instance, temporally-extended human activities are one example of complex concepts. For these, a more elaborate encoding and a more sophisticated clustering mechanism are needed as described in Chapter 3.

One of the key contributions we offer in this field is the use of incremental Gaussian mixture models (IGMM) and Bayesian information criterion (BIC) to learn the simple visual concepts in a loosely-supervised manner. The extended spatio-temporal graphs (STDAG) representation is also a key contribution of our work acting as an intermediary representation between the continuous perceptual space, and the purely symbolic linguistic structures.

7.1.2 Language grounding

The language grounding is the second step in our learning framework, and is performed after updating the visual concepts in each video clip. We ground natural language sentences to the learned concepts (e.g. colours, objects, people, human activities, etc.) in order to enable our robots to communicate effectively with the humans in their environment.

First, it is essential to acquire a natural language description of what the robot is perceiving to perform the grounding. Ideally we would like our robot to have a speech recognition modality and the capacity to ask people about particular objects or actions using natural language, but this remains a goal for future work. At present, we use Amazon Mechanical Turk and volunteers to collect multiple natural language descriptions of video clips recorded by each robot.

For grounding, we search for the highest correlations between words and phrases in a video clip description and the visual concepts that feature in that clip, allowing multi-to-multi associations to preserve the richness of natural language. The multi-to-multi association is enabled using integer programming. After finding the highest correlations, each is validated using our mental simulation idea which is enabled using graph matching technique. The use of integer programming along with mental simulation validation are the key contributions we offer in the field of language grounding.

7.1.3 Grammar induction

Grammar induction is the third and final step in our learning framework. Our grammar induction technique integrates with the previous two components by using the knowledge gained to enable learning of grammar rules. Using learned vision concepts and groundings to learn grammar strengthen our claims of cognitive plausibility, where a our robots learn about natural language and vision with minimum human supervision, allowing each robot to learn from its own experience.

We learn probabilistic grammar rules by mapping natural language commands to visual inputs. The main contribution we offer in the field of grammar induction is that we automatically generate training examples similar to those annotated by a human expert. This is achieved by exploiting the learned groundings and the extracted vision trees to successfully replace the human annotator. We formulate the automatic generation of language trees into a search problem. We search the space of all possible language trees from a sentence for one that matches the extracted vision tree. Given a match, we use that language tree to learn grammar. The procedure to perform the search is divided into five steps (substitute, group, query, match and learn) as presented in Algorithm 1.

7.2 Discussion

7.2.1 Loosely-supervised learning

We call our entire learning framework loosely-supervised even though the techniques used to learn about visual concepts and language components are unsupervised. For example, we use IGMM with BIC to model simple visual concepts and to select the number of components in an unsupervised manner. Also, we use LDA with Variational Bayes to learn and model complex human activities in an unsupervised way. Further, we use integer programming and graph matching to learn the language grounding from raw language and vision inputs without direct supervision. However, our framework is named loosely-supervised as opposed to unsupervised for three reasons: First, videos were manually segmented to include a single action in each. Second, videos and sentences were temporally aligned beforehand. Third, visual feature spaces (e.g. faces, shapes, etc.) were manually defined. We believe a fully unsupervised system should learn from long videos and text, i.e. be able to temporally segment the videos and map it to sentences automatically. Also, it should be able to generate new feature spaces when needed. This will allow our system to learn from much more rich sources like *YouTube* videos and *wikihow* descriptions. However, this remains as an ambition for future work.

7.2.2 Machine translation

On several occasions, the analogy between machine translation and our system has been mentioned. The analogy is between (A) how we learn the mapping between language and vision, and (B) how machine translation learns the mapping between two languages, e.g. learning translation in parliamentary proceedings with multiple language versions as presented by Koehn (2005). In other words, why not consider the vision domain to be another language and learn the translation between language and vision? Analysis suggests that indeed some aspects of machine translation techniques could be used to substitute parts of our system, and even further improve on it. For example, our system might be able to benefit from an idea used in machine translation bootstrapping to find the mapping between words in language and visual concepts. The bootstrapping idea explained in Knight (1999) explains how to use the Expectation Maximisation technique presented by Dempster *et al.* (1977) to find a suitable translation and alignment between the two languages. We are mainly interested in the translation part in this work. However, we believe learning the alignment too could help our system generate language descriptions for visual scenes, in particular longer scenes. For example, the work presented by Rohrbach *et al.* (2013; 2015) and Venugopalan *et al.* (2015) show how deep learning techniques can be used to learn to describe video snippets with natural language sentences. We leave these ideas open for future work to explore.

7.3 Future Work

Several research directions might emerge from our work; some improve on the existing framework, others build on it. Our approach suffers from two main limitations that hinder learning from longer videos, such as continuous stream of audio-video data or YouTube videos. First, it requires the videos and sentences to be temporally aligned beforehand, and second, it requires the feature spaces (e.g. colours, shapes, etc.) to be specified beforehand (though not their discretisation, which is learned). We discuss both points in detail in the following sections. We also discuss two main research directions that we believe are achievable by building on our existing system, which are (1) using the gained knowledge to enhance the learning from new language and vision data, and (2) learning in the presence of external knowledge.

7.3.1 Learning from non-segmented videos and text

Providing our robots with the ability to learn from long, non-segmented videos and text will significantly improve the learning. This will allow our robots to learn from rich web-available sources such as *YouTube* videos. Our language grounding and grammar induction frameworks are based on the idea that sentences map to their corresponding input videos, and having longer sentences and videos would break our assumption and prevent the learning. However, our system can be upgraded using an idea similar to that presented by Alayrac *et al.* (2016a). In their work, they presented a system capable of automatically learning the main steps to complete a given task, such as changing a car tire, from a set of narrated instruction videos. They addressed this task by formulating the problem as two clustering tasks, one in text and one in video, and then linking both domains by joint constraints. However, they assumed the language grammar is known, and used it to parse the long descriptions into smaller entities they called *direct object relations*, which consist of a single 'verb' and 'object' in each such as "remove tire".

7.3.2 Generating new visual features

Visual features are the representation or encoding we use to move from pixel level inputs into a space where visual concepts can be learned. These features are manually defined in this work, such as the HSL colour feature space. The manual definition of these feature spaces enables the robot to learn interesting concepts within the feature space, such the colours *red, green, blue, etc.* Automatically generating new visual feature spaces will enable our robot to learn more visual concept without the need to manually defining each feature space. One possible way of addressing this problem is to create a set of primitive features, which can be used to generate new feature space as presented by Bennett *et al.* (2016). In their paper, they addressed the problem of generating relational calculi from a set of primitive relations, as opposed to manually defining all relations in an ad hoc way. The work was limited to generating relations only, however, it can be expanded to include other features such as human activities, and object properties. By using this idea, we can reduce the problem of generating all possible feature spaces, into finding the set of primitive features that can be used to generate new visual features.

7.3.3 Using gained knowledge to improve learning

When learning a new language, the space of possible meanings of every word you hear is infinite. However, once the grammar rules are learned, an educated guess can be made to limit the space of possible meanings for each word. For example, provided that someone can speak English, if this person is provided with the sentence "*pick up the x block*". Assuming the person does not know what x means, s/he can guess that x is a property of the block, since the location of the word x is usually reserved for adjectives describing the following noun in this sentence. Using a similar concept, the gained knowledge in language grounding and grammar induction components can be used to influence the learning of new words, or bias the learning towards an expected outcome as presented in Figure 7.1.

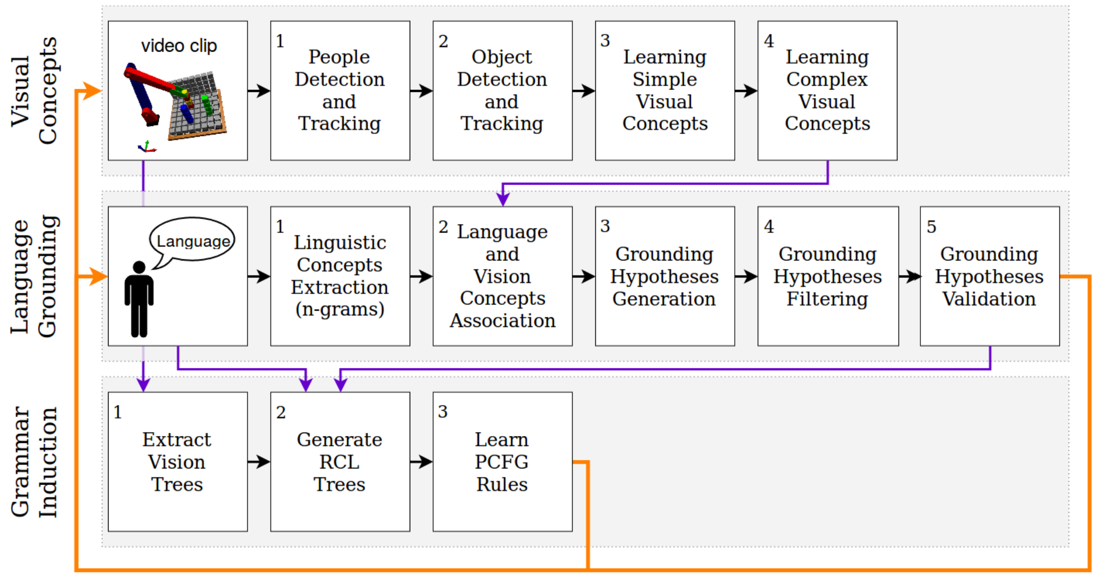

Visual Concepts
video clip
1 People Detection and Tracking
2 Object Detection and Tracking
3 Learning Simple Visual Concepts
4 Learning Complex Visual Concepts
Language Grounding
Language
1 Linguistic Concepts Extraction (n-grams)
2 Language and Vision Concepts Association
3 Grounding Hypotheses Generation
4 Grounding Hypotheses Filtering
5 Grounding Hypotheses Validation
Grammar Induction
1 Extract Vision Trees
2 Generate RCL Trees
3 Learn PCFG Rules

Figure 7.1: Extending our learning framework to use the gained knowledge in language grounding and grammar induction to influence the learning of new words and concepts, as indicated by the orange arrows.

7.3.4 Learning in the presence of external knowledge

This thesis focused on learning from robots' experiences only, where each robot learns about language and vision by manipulating objects and observing humans performing different tasks. One future direction can build on our system by combining the knowledge gained from experience along with readily available external knowledge sources such as web-available videos and instructions. The learning from experience achieved in this thesis can focus on learning simpler actions and concepts to bootstrap the robot's knowledge (e.g. pick up and move objects). Once the bootstrapping is achieved, the robot can move to learn from rich external knowledge sources that contain more complex actions (e.g. cooking pasta) and require some background knowledge encoded in the robot. Using both knowledge sources in this order (learning from experience then external sources) is similar to how humans learn to perform different tasks. For example, a child learns to manipulate blocks and toys before learning how to make a sandwich or cut the grass. We believe this learning architecture is more capable of achieving human-level performance as the robot learns the basics about language and vision from its own experience, and is capable of learning more basic concepts when needed, as opposed to having to program each basic concept individually for each task.

Appendix A

Incremental Gaussian Mixture Models Approach

The Incremental Gaussian Mixture Model (IGMM) approach used in this thesis was first presented by Song and Wang (2005). In their paper, a new probability-density-based data clustering technique was presented, which requires only the newly observed data to be saved in memory, as opposed to storing the entire historical data. Their IGMM approach incrementally updates the density estimate by processing only the newly observed data, and having access to the previously estimated Gaussian components. The approach was implemented using the Expectation Maximisation (EM) algorithm along with a cluster merging strategy that deploys multivariate statistical tests to check for equality of covariance and mean. This approach is highly efficient in computational and memory requirements when clustering large amounts of online data streams if compared with the standard EM algorithm, which is an essential requirement in our work of life-long learning of language and vision for robotic agents.

A.1 Introduction

The data stream clustering problem is defined by Guha *et al.* (2003) as "to maintain a consistently good clustering of the sequence observed so far, using a small amount of memory and time". This definition emphasise on limited time and memory requirements relatively to the amount of data being clustered. Time-critical applications such as neuronal signal monitoring require real-time clustering of relatively small amounts of data. Memory-critical applications such as clustering financial transactions over a period of years, require massive data clustering. Other applications are both time and memory critical such as the work presented in this thesis of teaching a robot about language and vision in an incremental manner.

In their paper, Song and Wang use the following terminologies to describe the different data types used in the paper, we follow the same terminology in this section. They define "recent data" as all data available in the memory from a data stream, while they define "historical data" as all data received in the data stream so far, this include the recent data. Historical data is not stored in memory, except for the recent data portion. I.e. only recent data is stored in memory. Also, they call unprocessed recent data "newly arrived data". If the entire historical data were available in memory, then the Gaussian mixture model (GMM) technique would have been effectively employed to estimate the Gaussian components and cluster the data using the EM algorithm. However, this is not the case in this thesis, as we assume the robot does not store all of the observed data in the memory. Also, humans do not re-evaluate their entire knowledge base (i.e. re-cluster) whenever they observe a new object or activity. That is why we try to perform the clustering in an online-incremental manner. The EM algorithm (or any of its known variations) is not applicable for a data stream without complete historical records available in memory. Song and Wang argue in their paper that their new IGMM technique can adapt probability density based clustering algorithms to solve data stream clustering problems much more efficiently than applying the EM on the entire historical data. This is achieved by applying the standard EM algorithm only on newly arrived data, while the IGMM estimation algorithm merges newly found Gaussian components with previously learned ones that are statistically equivalent as shown in Figure 3.16 in this thesis. The statistical equivalence of any two Gaussian components is determined using two tests: the W-statistic test to check for equality of covariances, and the Hotelling's-T^2 statistic test to check for equality of mean. The sufficient statistics of mean and covariance for a multivariate normal distribution make it possible to perform the tests and merging without resort to historical data, i.e. to cluster a stream of data by having access to only recent data and the learned Gaussian components.

In this section, we follow the notation presented in Song and Wang's (2005) paper. Let T be the discrete time when the data point x_T in $\mathbb{R}^d$ is observed, while d represents the number of dimensions which vary across feature spaces in this thesis. For example, the HSL feature space has 3 dimensions, while FPFH has 32. In their paper, x_T is regarded as a random vector, while we consider it to be the observed values in a visual feature space in a single input video clip. Let $g^T : \mathbb{R}^d \to \mathbb{R}$ be an estimator of the true probability density function (p.d.f.) $p_0(x)$ based on the data points observed from time 1 to T. Let $g^N(x)$ be an estimator of $p_0(x)$ based on the historical data $x_1, \ldots, x_N$. Let $a(x)$ be an estimator of $p_0(x)$ based on the newly arrived data $x_{N+1}, \ldots, x_{N+M}$. Let $g^{N+M}(x)$ be an estimator of $p_0(x)$ based on both the historical data $x_1, \ldots, x_N$ and the newly arrived data $x_{N+1}, \ldots, x_{N+M}$. The data stream clustering problem that we will address in this appendix is: obtain the estimator $g^{N+M}(x)$ from $g^N(x)$ and the M newly arrived data sample $x_{N+1}, \ldots, x_{N+M}$, i.e. to estimate $g^{N+M}(x)$ from the previously learned Gaussian components and the new observation.

A.2 Updating A Gaussian Mixture Model

The p.d.f. of a GMM is written as:

$$\sum_{k=1}^{K} \pi_k \phi(x|\mu_k, \Sigma_k)$$

with

$$\sum_{k=1}^{K} \pi_k = 1,\ 0 \le \pi_k \le 1,\ \text{for } k = 1, \ldots, K$$

where $\phi(x|\mu, \Sigma)$ is the p.d.f. of a multivariate normal distribution with mean vector μ and covariance matrix Σ:

$$\phi(x|\mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right)$$

We represent $g^N(x)$ by a GMM with parameters:

$$\pi_j, \mu_j, \Sigma_j, j = 1, \ldots, K_g$$

and $a(x)$ by a GMM with parameters:

$$\pi_k, \mu_k, \Sigma_k, k = 1, \ldots, K_a$$

where K_g and K_a are the numbers of components in each GMM respectively.

To update the Gaussian mixture model, we first obtain a GMM for $a(x)$ from the newly arrived data $x_{N+1}, \ldots, x_{N+M}$ with K_a components. The number of components is selected unsupervised using the Bayesian Information Criterion (BIC) technique. According to the GMM of $a(x)$, the newly arrived data $x_{N+1}, \ldots, x_{N+M}$ is separated into K_a clusters. Let D_k be the collection of data in cluster k. Let M_k be the number of data points in D_k. For each cluster k, we will determine if it has a statistically equivalent covariance, using W-statistic test, and mean, using Hotelling's T^2 statistic test, with any of the Gaussian components in $g^N(x)$. I.e. we will test to see if any of the newly clustered Gaussian components in $a(x)$ matches with any of the previously learned Gaussian components in $g^N(x)$. This translates to mapping newly found vision concepts to previously observed and learned ones in this thesis.

The mean equality test happens after covariance equality test as the Hotelling's T^2 test assumes equality of covariances between components. If any component j in $g^N(x)$ is found to be equivalent to the component k in $a(x)$, then we create a new component in $g^{N+M}(x)$ by merging both components j and k together using Equations (A.1, A.2, A.3). If not, we will add the component k of $a(x)$ as a new component to $g^{N+M}(x)$ with an adjusted weight using Equation (A.4). All remaining components in $g^N(x)$ will be added to $g^{N+M}(x)$ with adjusted weights too using Equation (A.5). At the end, we perform both statistical tests one more time on all components of $g^{N+M}(x)$ to merge statistically equivalent components with a similar strategy. The overall algorithm is presented in Algorithm 2 which was initially presented by Song and Wang (2005).

A.2.1 Merging or creating components

If the new data points D^k passes both statistical tests (covariance and mean) for component j of $g^N(x)$, then we consider component k of $a(x)$ and component j to be statistically equivalent. We merge the two to create a new component in $g^{N+M}(x)$ with mean μ, covariance matrix Σ

Algorithm 2 Incremental Gaussian Mixture Model Estimation

```
1: procedure
2: Input GMM $g^N(x)$, newly arrived data $x_{N+1}, \dots, x_{N+M}$
3: Output GMM $g^{N+M}(x)$
4: Perform EM algorithm to estimate the Gaussian mixture model $a(x)$ from the new data
   $x_{N+1}, \dots, x_{N+M}$, with number of components $Ka$ determined by BIC.
5: Assign each new data $x_m$ to the most likely component according to the conditional prob-
   ability
                              $Prob(k|_x m), k = 1, \dots, K_a$
6:     for each component $k$ in $a(x)$ do
7:         Let $D^k$ be the collection of all data in component $k$
8:         for component $j$ with mean $\mu_j$ and covariance $\Sigma_j$ in $g^N(x)$ do
9:             Calculate the W-statistic to determine if $D^k$ has equal covariance with $\Sigma_j$
10:            if $D^k$ has passed the covariance test then
11:                Perform the Hotelling's $T^2$ test to determine if $D^k$ has the same mean $\mu_j$
12:                if $D^k$ has passed the mean test then
13:                    Consider components $k$ in $a(x)$ and $j$ in $g^N(x)$ identically distributed
14:                    Compute the log likelihood of component $j$ for $D^k$ to break possible ties
15:    for each pair of equivalent components in $g^N(x)$ and $a(x)$ do
16:        Create a new component in $g^{N+M}(x)$ by merging the pair using Eq. (A.1, A.2, A.3)
17:    for each remaining component $k$ in $a(x)$ do
18:        Assign this component to $g^{N+M}(x)$ with an updated weight using Equation (A.4)
19:    for each remaining component $j$ in $g^N(x)$ do
20:        Assign this component to $g^{N+M}(x)$ with an updated weight using Equation (A.5)
21:    Merge statistically equivalent components in $g^{N+M}(x)$
22:    Return $g^{N+M}(x)$
```

and weight π using Equations (A.1, A.2, A.3).

$$\mu = \frac{N\pi_j\mu_j + M_k\mu_k}{N\pi_j + M_k} \tag{A.1}$$

$$\Sigma = \frac{N\pi_j\Sigma_j + M_k\Sigma_k}{N\pi_j + M_k} + \frac{N\pi_j\mu_j\mu_j^\top + M_k\mu_k\mu_k^\top}{N\pi_j + M_k} - \mu\mu^\top \tag{A.2}$$

$$\pi = \frac{N\pi_j + M_k}{N + M} \tag{A.3}$$

For each component k in $a(x)$ that does not have a statistically equivalent component in $g^N(x)$, we create a new component in $g^{N+M}(x)$ with mean and covariance equal to that of component k, i.e. $\mu = \mu_k$ and $\Sigma = \Sigma_k$, but with a weight presented in Equation (A.4).

$$\pi = \frac{M_k}{N + M} \tag{A.4}$$

For each remaining component j in $g^N(x)$ that does not have a statistically equivalent component in $a(x)$, we create a new component in $g^{N+M}(x)$ with mean and covariance equal to that of component j, i.e. $\mu = \mu_j$ and $\Sigma = \Sigma_j$,, but with a weight presented in Equation (A.5).

$$\pi = \frac{N\pi_j}{N+M} \tag{A.5}$$

References

S. Abney. Partial parsing via finite-state cascades. *Natural Language Engineering*, 2(04):337–344, 1996. 130

J.-B. Alayrac, P. Bojanowski, N. Agrawal, J. Sivic, I. Laptev, and S. Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4575–4583, 2016a. 142

J.-B. Alayrac, P. Bojanowski, N. Agrawal, J. Sivic, I. Laptev, and S. Lacoste-Julien. Unsupervised learning from narrated instruction videos. 2016b. 84

J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983. viii, 27, 44, 50

M. Alomari, E. Chinellato, Y. Gatsoulis, D. C. Hogg, and A. G. Cohn. Unsupervised Grounding of Textual Descriptions of Object Features and Actions in Video. In *15th International Conference on Principles of Knowledge Representation and Reasoning*, 2016a. 30, 62

M. Alomari, P. Duckworth, Y. Gatsoulis, D. C. Hogg, and A. G. Cohn. Unsupervised natural language acquisition and grounding to visual representations for robotic systems. In *Workshop on Cognitive Knowledge Acquisition and Applications (Cognitum 2016), IJCAI 2016.*, 2016b. 62

M. Alomari, P. Duckworth, N. Bore, M. Hawasly, D. C. Hogg, and A. G. Cohn. Grounding of human environments and activities for autonomous robots. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2017a. x, 27, 51, 55, 62, 110

M. Alomari, P. Duckworth, D. C. Hogg, and A. G. Cohn. Natural language acquisition and grounding for embodied robotic systems. In *Proceedings of Association for the Advancement of Artificial Intelligence (AAAI)*. AAAI Press, 2017b. 26, 62, 103

M. Alomari, P. Duckworth, D. C. Hogg, and A. G. Cohn. Learning of object properties, spatial relations, and actions for embodied agents from language and vision. In *Proceedings of Interactive Multi-Sensory Object Perception for Embodied Agents (AAAI Spring Symposium)*, 2017c. 62

D. P. Barrett, S. A. Bronikowski, H. Yu, and J. M. Siskind. Robot language learning, generation, and comprehension. *arXiv preprint arXiv:1508.06161*, 2015. 62

D. P. Barrett, S. A. Bronikowski, H. Yu, and J. M. Siskind. Driving under the influence (of language). *IEEE Transactions on Neural Networks and Learning Systems*, 2017. 14

P. W. Battaglia, J. B. Hamrick, and J. B. Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45):18327–18332, 2013. 72

M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mosenlechner, D. Pangercic, T. Ruhr, and M. Tenorth. Robotic roommates making pancakes. In *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, pages 529–536. IEEE, 2011. 13, 84

A. Behera, A. G. Cohn, and D. C. Hogg. Workflow activity monitoring using dynamics of pair-wise qualitative spatial relations. In *International Conference on Multimedia Modeling*, pages 196–209. Springer, Berlin, Heidelberg, 2012. 24, 26

B. Bennett, H. Du, L. Gomez Alvarez, and A. G. Cohn. Defining relations: a general incremental approach with spatial temporal case studies. In *Frontiers in Artificial Intelligence and Applications*, volume 263, pages 23–36. IOS Press, 2016. 142

T. L. Berg, A. C. Berg, J. Edwards, M. Maire, R. White, Y.-W. Teh, E. Learned-Miller, and D. A. Forsyth. Names and faces in the news. In *Computer Vision and Pattern Recognition. CVPR. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–II. IEEE, 2004. 22, 34

B. Berlin and P. Kay. *Basic color terms: their universality and evolution.* Berkeley: University of California Press, 1969. 3, 36

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of ML Research*, 3, 2003. 33, 51

J. Bleys, M. Loetzsch, M. Spranger, and L. Steels. The grounded colour naming game. *Proceedings of Spoken Dialogue and Human-Robot Interaction Workshop at the RoMan 2009 Conference*, 2009. 11

N. Bore, R. Ambrus, P. Jensfelt, and J. Folkesson. Efficient retrieval of arbitrary objects from long-term robot observations. *Robotics and Autonomous Systems*, 2017. 32, 33, 109

A. Boularias, F. Duvallet, J. Oh, and A. Stentz. Learning qualitative spatial relations for robotic navigation. In *IJCAI*, pages 4130–4134, 2016. 25

M. E. Bouton. *Learning and behavior: A contemporary synthesis.* Sinauer Associates, 2007. 77

M. Bowerman. The origins of childrens spatial semantic categories: Cognitive versus linguistic determinants. *Rethinking linguistic relativity*, pages 145–176, 1996. 3, 36, 39

S. Calinon and A. Billard. Incremental learning of gestures by imitation in a humanoid robot. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 255–262. ACM, 2007. 25

J. B. Carroll. Language, thought, and reality: Selected writings of Benjamin Lee Whorf. 1956. 3

D. L. Chen and R. J. Mooney. Learning to interpret natural language navigation instructions from observations. In *Proceedings of Association for the Advancement of Artificial Intelligence (AAAI)*, volume 2, pages 1–2. AAAI Press, 2011. 17, 62

J. Chen, A. G. Cohn, D. Liu, S. Wang, J. Ouyang, and Q. Yu. A survey of qualitative spatial representations. *The Knowledge Engineering Review*, pages 1–31, 2013. 23

N. Christofides. *Graph Theory An Algorithmic Approach.* New York: Academic Press Inc., 1975. 44

E. Clementini, P. Di Felice, and D. Hernández. Qualitative representation of positional information. *Artificial Intelligence*, 95(2):317 – 356, 1997. 23, 50

A. G. Cohn and N. M. Gotts. The ‘egg-yolk’ representation of regions with indeterminate boundaries. In P. Burrough and A. M. Frank, editors, *Proceedings, GISDATA Specialist Meeting on Geographical Objects with Undetermined Boundaries*, pages 171–187. Francis Taylor, 1996. viii, 44

M. Collins. Three generative, lexicalised models for statistical parsing. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 16–23. Association for Computational Linguistics, 1997. 18

T. M. Cover and J. A. Thomas. Elements of information theory. 1991. 99, 113

K. J. W. Craik. *The nature of explanation*, volume 445. CUP Archive, 1967. 71, 121

M. Delafontaine, A. G. Cohn, and N. Van de Weghe. Implementing a qualitative calculus to analyse moving point objects. *Expert Systems with Applications*, 38(5):5187 – 5196, 2011. 24, 50

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977. 141

P. F. Dominey and J. D. Boucher. Developmental Stages of Perception and Language Acquisition in a Perceptually Grounded Robot. *Cognitive Systems Research*, 6(3):243–259, 2005. 83

K. S. Dubba, M. R. De Oliveira, G. H. Lim, H. Kasaei, L. S. Lopes, A. Tomé, and A. G. Cohn. Grounding Language in Perception for Scene Conceptualization in Autonomous Robots. In *Qualitative Representations for Robots: Papers from the AAAI Spring Symposium. Technical report*, pages 26–33, 2014. 13

P. Duckworth, M. Alomari, Y. Gatsoulis, D. C. Hogg, and A. G. Cohn. Unsupervised activity recognition using latent semantic analysis on a mobile robot. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI 2016)*, August 2016a. 33, 51

P. Duckworth, Y. Gatsoulis, F. Jovan, D. C. Hogg, and A. G. Cohn. Unsupervised learning of qualitative motion behaviours by a mobile robot. In *Proceedings of the 15th Int Conf on Autonomous Agents & Multiagent Systems (AAMAS) 2016*, 2016b. 27, 44

P. Duckworth, M. Alomari, J. Charles, D. C. Hogg, and A. G. Cohn. Latent dirichlet allocation for unsupervised activity analysis on an autonomous mobile robot. In *Proceedings of Association for the Advancement of Artificial Intelligence (AAAI)*, 2017. xiii, 27, 33, 49, 50, 51, 110

K. Dukes. Train Robots: A Dataset for Natural Language Human-Robot Spatial Interaction through Verbal Commands. In *International Conference on Social Robotics (ICSR). Embodied Communication of Goals and Intentions Workshop*, 2013. xii, 81, 82, 86, 88, 100, 101

K. Dukes. Semeval-2014 task 6: Supervised semantic parsing of robotic spatial commands. *SemEval 2014*, page 45, 2014. 16, 62, 83, 84

B. S. Everitt and A. Skrondal. The Cambridge dictionary of statistics. *University of Cambridge Press: Cambridge*, 2002. 65

K. Fukui and O. Yamaguchi. Face recognition using multi-viewpoint patterns for robot vision. In *Robotics Research. The Eleventh International Symposium*, pages 192–201. Springer, 2005. 22, 34

A. Galata, A. Cohn, D. Magee, and D. Hogg. Modeling interaction using learnt qualitative spatio-temporal relations and variable length markov models. In *Proceedings of the 15th European Conference on Artificial Intelligence*, pages 741–745. IOS Press, 2002. 24

Y. Gatsoulis, M. Alomari, C. Burbridge, C. Dondrup, P. Duckworth, P. Lightbody, M. Hanheide, N. Hawes, and A. G. Cohn. QSRlib: a software library for online acquisition of Qualitative Spatial Relations from Video. In *Workshop on Qualitative Reasoning, at IJCAI*, 2016a. 23, 44, 50

Y. Gatsoulis, P. Duckworth, C. Dondrup, P. Lightbody, and C. Burbridge. QSRlib: A library for qualitative spatial-temporal relations and reasoning. qsrlib.readthedocs.org, Jan 2016b. 50

A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian data analysis*, volume 2. Chapman & Hall/CRC Boca Raton, 2014. 118

goslate 1.5.1. https://pypi.python.org/pypi/goslate, January 2016. 102, 125

S. Griffith, J. Sinapov, M. Miller, and A. Stoytchev. Toward interactive learning of object categories by a robot: A case study with container and non-container objects. In *Development and Learning, 2009. ICDL 2009. IEEE 8th International Conference on*, pages 1–6. IEEE, 2009. 21

S. Guadarrama, L. Riano, D. Golland, D. Go, Y. Jia, D. Klein, P. Abbeel, T. Darrell, et al. Grounding spatial relations for human-robot interaction. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1640–1647. IEEE, 2013. 12

S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams: Theory and practice. *IEEE transactions on knowledge and data engineering*, 15(3):515–528, 2003. 145

S. Hemachandra, F. Duvallet, T. M. Howard, N. Roy, A. Stentz, and M. R. Walter. Learning models for following natural language directions in unknown environments. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 5608–5615. IEEE, 2015. 14

M. Hoffman, F. Bach, and D. Blei. Online learning for latent dirichlet allocation. In *Advances in neural information processing systems*, 2010. 55

D. C. Hogg. A methodology for real time scene analysis. In *IJCAI*, volume 2, page 627, 1977. 9

E. Howorka. A characterization of distance-hereditary graphs. *The quarterly journal of mathematics*, 28(4):417–420, 1977. 74

Y. Hristov, S. Penkov, A. Lascarides, and S. Ramamoorthy. Grounding symbols in multi-modal instructions. *Proceedings of the ACL workshop Language Grounding for Robotics*, 7 2017. 12

A. S. Huang, S. Tellex, A. Bachrach, T. Kollar, D. Roy, and N. Roy. Natural language command of an autonomous micro-air vehicle. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2663–2669. IEEE, 2010. 13

C. L. Hudson Kam and E. L. Newport. Regularizing unpredictable variation: The roles of adult and child learners in language formation and change. *Language Learning and Development*, 1(2):151–195, 2005. 16

M. Kaboli, P. Mittendorfer, V. Hügel, and G. Cheng. Humanoids learn object properties from robust tactile feature descriptors via multi-modal artificial skin. In *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*, pages 187–192. IEEE, 2014. 21

U. Klank, D. Pangercic, R. B. Rusu, and M. Beetz. Real-time CAD Model Matching for Mobile Manipulation and Grasping. In *9th IEEE-RAS International Conference on Humanoid Robots*, pages 290–296, Paris, France, December 7-10 2009. 32, 104

K. Knight. A statistical mt tutorial workbook. In *Prepared for the 1999 JHU Summer Workshop*, 1999. 141

P. Koehn. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86, 2005. 141

L. Kunze, C. Burbridge, and N. Hawes. Bootstrapping probabilistic models of qualitative spatial relations for active visual object search. In *AAAI Spring Symposium*, pages 24–26, 2014. 24

K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer speech & language*, 4(1):35–56, 1990. 93, 129

S. Lauria, G. Bugmann, T. Kyriacou, and E. Klein. Mobile robot programming using natural language. *Robotics and Autonomous Systems*, 38(3):171–181, 2002. 13, 62

R. Lienhart and J. Maydt. An extended set of haar-like features for rapid object detection. In *Image Processing*, 2002. 35

M. MacMahon, B. Stankiewicz, and B. Kuipers. Walk the talk: Connecting language, knowledge, and action in route instructions. *AAAI*, 2(6):4, 2006. 16

J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967. 24

P. C. Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)*, 2:49–55, 1936. 42

J. Malmaud, J. Huang, V. Rathod, N. Johnston, A. Rabinovich, and K. Murphy. What's cookin'? interpreting cooking videos using text, speech and vision. *arXiv preprint arXiv:1503.01558*, 2015. 84

C. Matuszek, E. Herbst, L. Zettlemoyer, and D. Fox. Learning to parse natural language commands to a robot control system. In *Experimental Robotics*, pages 403–415. Springer, 2013. 14, 16, 62, 83, 84

F. Mery and T. J. Kawecki. An operating cost of learning in drosophila melanogaster. *Animal Behaviour*, 68(3):589–598, 2004. 7

MetraLabs. www.metralabs.com/en, 2016. 29

Microsoft. Xtion Pro Live. *URL https://www.asus.com/3D-Sensor/Xtion_PRO_LIVE*, 2014. 30

Microsoft. "Kinect for Windows features", (Last accessed 16-12-2016). `http://www.microsoft.com/en-us/kinectforwindows/meetkinect/features.aspx`, 2015. 30, 103

D. K. Misra, J. Sung, K. Lee, and A. Saxena. Tell me Dave: Context-sensitive grounding of natural language to manipulation instructions. *JAIR*, page 0278364915602060, 2015. 62

D. Moore and I. Essa. Recognizing multitasked activities from video using stochastic context-free grammar. In *AAAI/IAAI*, pages 770–776, 2002. 18

R. Moratz and M. Ragni. Qualitative spatial reasoning about relative point position. *Journal of Visual Languages & Computing*, 19(1):75–98, 2008. 23, 50

M. Muja and M. Ciocarlie. (last accessed 16-12-2016). http://www.ros.org/wiki/tabletop_object_detector, 2013. 32, 104

C. J. Needham, P. E. Santos, D. R. Magee, V. Devin, D. C. Hogg, and A. G. Cohn. Protocols from perceptual observations. *Artificial Intelligence*, 167(1):103–136, 2005. 2, 11

H. G. Nguyen, J. Morrell, K. D. Mullens, A. B. Burmeister, S. Miles, N. Farrington, K. M. Thomas, and D. W. Gage. Segway robotic mobility platform. In *Optics East*, pages 207–220. International Society for Optics and Photonics, 2004. 29, 106

NLTK: Hidden Markov Model (HMM) for Part-of-Speech (POS) tagging. http://www.nltk.org/_modules/nltk/tag/hmm.html, 2017. 123

OpenNI. www.openni.org, 2016. 31, 57, 109

R. E. Owens Jr. Language development: An introduction— edition: 9. *Instructor*, 2016. 65

G. Paolacci, J. Chandler, and P. G. Ipeirotis. Running experiments on Amazon Mechanical Turk. *Judgment and Decision making*, 5(5):411–419, 2010. 100

J. Papon, A. Abramov, M. Schoeler, and F. Worgotter. Voxel cloud connectivity segmentation-supervoxels for point clouds. In *CVPR*, 2013. 32

N. Parde, A. Hair, M. Papakostas, K. Tsiakas, M. Dagioglou, V. Karkaletsis, and R. D. Nielsen. Grounding the Meaning of Words through Vision and Interactive Gameplay. In *Proceedings of International Joint Conference on Artificial Intelligence(IJCAI)*. AAAI Press, 2015. 12

O. M. Parkhi, A. Vedaldi, A. Zisserman, et al. Deep face recognition. In *BMVC*, volume 1, page 6, 2015. 22

P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal. Learning and generalization of motor skills by learning from demonstration. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 763–768. IEEE, 2009. 26

R. Pfeifer and C. Scheier. Sensorymotor coordination: The metaphor and beyond. *Robotics and autonomous systems*, 20(2-4):157–178, 1997. 20

H. Pfister, M. Zwicker, J. Van Baar, and M. Gross. Surfels: Surface elements as rendering primitives. In *Computer Graphics and Interactive Techniques*, 2000. 31

J. Piaget. *The construction of reality in the child*. Basic Books, 1954. 3, 36, 39

E. Ponvert, J. Baldridge, and K. Erk. Simple unsupervised grammar induction from raw text with cascaded finite state models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. 17, 82, 130, 133

D. Posada and T. R. Buckley. Model selection and model averaging in phylogenetics: advantages of akaike information criterion and Bayesian approaches over likelihood ratio tests. *Systematic biology*, 53(5):793–808, 2004. 35

M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. ROS: an open-source Robot Operating System. *Workshop on open source software (at ICRA)*, 3 (3.2):5, 2009. 29

M. Quine. Word and object. 1960. 10, 61

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. 122

K. Ramirez-Amaro, M. Beetz, and G. Cheng. Transferring skills to humanoid robots by extracting semantic representations from observations of human activities. *Artificial Intelligence*, 247:95–118, 2015. 26

R. Robotics. Baxter. *Retrieved Jan*, 10:2014, 2013. 29

Robotnik. http://www.robotnik.eu/robotics-arms/kinova-mico-arm/, 2014. 29, 106

A. Rohrbach, M. Rohrbach, N. Tandon, and B. Schiele. A dataset for movie description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3202–3212, 2015. 141

M. Rohrbach, W. Qiu, I. Titov, S. Thater, M. Pinkal, and B. Schiele. Translating video content to natural language descriptions. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2013. 141

E. Rosch and C. B. Mervis. Children's sorting: A reinterpretation based on the nature of abstraction in natural categories. *Readings in child development and relationships*, pages 140–148, 1977. 3, 36

A. Rosenberg and J. Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*, volume 7, pages 410–420, 2007. 99, 113

B. Rosman and S. Ramamoorthy. Learning spatial relationships between objects. *The International Journal of Robotics Research*, 30(11):1328–1342, 2011. 24

D. Roy, B. Schiele, and A. Pentland. Learning Audio-Visual Associations using Mutual Information. In *Integration of Speech and Image Understanding, 1999. Proceedings*, pages 147–163. IEEE, 1999. 2, 11, 20, 62

R. B. Rusu. *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*. PhD thesis, Computer Science department, Technische Universitaet Muenchen, Germany, 2009. 37

R. B. Rusu. Semantic 3d object maps for everyday manipulation in human living environments. *KI-Künstliche Intelligenz*, 24(4):345–348, 2010. 37

R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011. 104

R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz. Towards 3d point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941, 2008. 20

R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2155–2162. IEEE, 2010. 37

C. J. Schatz. The Developing Brain. *Scientific American*, 267(3):60–67, 1992. 65

M. Schoeler, J. Papon, and F. Worgotter. Constrained planar cuts-object partitioning for point clouds. In *CVPR*, 2015. 31

D. A. Schult and P. Swart. Exploring network structure, dynamics, and function using networkx. In *Proceedings of the 7th Python in Science Conferences (SciPy 2008)*, volume 2008, pages 11–16, 2008. 55

C. Shain, W. Bryce, L. Jin, V. Krakovna, F. Doshi-Velez, T. A. Miller, W. Schuler, and L. Schwartz. Memory-bounded left-corner unsupervised grammar induction on child-directed input. In *COLING*, pages 964–975, 2016. 18

L. She, S. Yang, Y. Cheng, Y. Jia, J. Y. Chai, and N. Xi. Back to the blocks world: Learning new actions through situated human-robot dialogue. In *15th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, volume 89, 2014. 12, 62

S. P. Shelov and R. E. Hannemann. *Caring for Your Baby and Young Child: Birth to Age 5. The Complete and Authoritative Guide.* ERIC, 1993. 20, 36

J. Sinapov, P. Khante, M. Svetlik, and P. Stone. Learning to order objects using haptic and proprioceptive exploratory behaviors. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, 2016. xiii, 21, 29, 106

J. M. Siskind. A Computational Study of Cross-Situational Techniques for Learning Word-to-Meaning Mappings. *Cognition*, 61(1):39–91, 1996. 2, 11, 62

J. M. Siskind, J. J. Sherman Jr, I. Pollak, M. P. Harper, and C. A. Bouman. Spatial random tree grammars for modeling hierarchal structure in images with regions of arbitrary shape. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(9):1504–1519, 2007. 18

K. Sjöö and P. Jensfelt. Learning spatial relations from functional simulation. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 1513–1519. IEEE, 2011. 24

R. Socher, C. C. Lin, C. Manning, and A. Y. Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 129–136, 2011. 18

A. Søgaard. Unsupervised dependency parsing without training. *Natural Language Engineering*, 18(02):187–203, 2012. 17

M. Song and H. Wang. Highly Efficient Incremental Estimation of Gaussian Mixture Models for Online Data Stream Clustering. In *Defense and Security*, pages 174–183. International Society for Optics and Photonics, 2005. 52, 145, 147, 148

K. Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21, 1972. 80

M. Spranger. Incremental Grounded Language Learning in Robot-Robot Interactions - Examples from Spatial Language. In *Joint IEEE International Conferences on Development and Learning and Epigenetic Robotics (ICDL-Epirob)*, pages 196–201. IEEE, 2015. 3

M. Spranger and L. Steels. Co-acquisition of syntax and semantics - an investigation in spatial language. In Q. Yang and M. Wooldridge, editors, *Proceedings of International Joint Conference on Artificial Intelligence(IJCAI)*, pages 1909–1905, Palo Alto, US, 2015. AAAI Press. 12, 62

M. Sridhar, A. G. Cohn, and D. C. Hogg. Learning functional object categories from a relational spatio-temporal representation. In *ECAI 2008: 18th European Conference on Artificial Intelligence (Frontiers in Artificial Intelligence and Applications)*, pages 606–610. IOS Press, 2008. 27

M. Sridhar, A. G. Cohn, and D. C. Hogg. Discovering an event taxonomy from video using qualitative spatio-temporal graphs. In *ECAI 2010-19th European Conference on Artificial Intelligence, Proceedings*, volume 215, pages 1103–1104. IOS Press, 2010a. 44

M. Sridhar, A. G. Cohn, and D. C. Hogg. Unsupervised Learning of Event Classes from Video. In *Proceedings of Association for the Advancement of Artificial Intelligence (AAAI)*, pages 1631–1638. AAAI Press, 2010b. 44

M. Steedman. *The syntactic process*, volume 24. MIT Press, 2000. 16

L. Steels. Language Games for Autonomous Robots. *Intelligent Systems, IEEE*, 16(5):16–22, 2001. 11, 62

L. Steels and F. Kaplan. Aibo's First Words: The Social Learning of Language and Meaning. *Evolution of Communication*, 4(1):3–32, 2002. 3, 11

STRANDS. strands-project.eu, 2016. 29, 51, 55, 59, 108

J. Tayyub, A. Tavanai, Y. Gatsoulis, A. G. Cohn, and D. C. Hogg. Qualitative and quantitative spatio-temporal relations in daily living activity recognition. In *Asian Conference on Computer Vision*, pages 115–130. Springer, Cham, 2014. 27

S. Tellex, T. Kollar, S. Dickerson, M. R. Walter, A. G. Banerjee, S. Teller, and N. Roy. Approaching the symbol grounding problem with probabilistic graphical models. *AI magazine*, 32(4):64–76, 2011. 13, 62, 83, 84

J. Thomason, S. Zhang, R. J. Mooney, and P. Stone. Learning to interpret natural language commands through human-robot dialog. In *IJCAI*, pages 1923–1929, 2015. 14

C. Turati, V. Macchi Cassia, F. Simion, and I. Leo. Newborns' face recognition: Role of inner and outer facial features. *Child development*, 77(2):297–311, 2006. 34

A. M. Turing. Computing machinery and intelligence. *Mind*, 59(236):433–460, 1950. 1

M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1), 1991. 35

C. Van Rijsbergen. Information retrieval. dept. of computer science, university of glasgow. *URL: citeseer. ist. psu. edu/vanrijsbergen79information. html*, 14, 1979. 98, 122

S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko. Translating videos to natural language using deep recurrent neural networks. *Proceedings of the conference on European chapter of the Association for Computational Linguistics*, 2015. 141

S. I. Wang, P. Liang, and C. D. Manning. Learning language games through interaction. In *ACL*, 2016. 17

S. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *CVPR*, 2016. 31, 109

S. Wermter and M. Elshaw. Learning robot actions based on self-organising language memory. *Neural Networks*, 16(5):691–699, 2003. 14

Willowgarage. http://www.willowgarage.com/, 2010. 12

T. Winograd. Understanding natural language. *Cognitive Psychology*, 3(1):1–191, 1972. vii, 7, 9, 10

Y. Yang, Y. Li, C. Fermüller, and Y. Aloimonos. Robot learning manipulation action plans by. In *Watching Unconstrained Videos from the World Wide Web, in Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. 18

D. H. Younger. Recognition and parsing of context-free languages in time n3. *Information and control*, 10(2):189–208, 1967. 131