Article:
Alomari, M, Li, F, Hogg, DC orcid.org/0000-0002-6125-9564 et al. (1 more author) (2022) Online perceptual learning and natural language acquisition for autonomous robots. Artificial Intelligence, 303. 103637. p. 103637. ISSN 0004-3702

https://doi.org/10.1016/j.artint.2021.103637

Online Perceptual Learning and Natural Language Acquisition for Autonomous Robots

Muhannad Alomari[a], Fangjun Li[a], David C. Hogg[a], Anthony G. Cohn[a,b,c,d,*]

[a]*School of Computing, University of Leeds, UK*
[b]*Luzhong Institute of Safety, Environmental Protection Engeering and Materials, Qingdao University of Science and Technology, Zibo, 255000, China*
[c] *College of Electronic and Information Engineering, Tongji University, China*
[d]*School of Mechanical and Electrical Engineering, Qingdao University of Science and Technology, China*
[e]*School of Control Science and Engineering, Shandong University, Jinan, 250061, China*

Abstract

In this work, 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. In particular, we demonstrate a system, called OLAV, which is able, for the first time, to (1) learn to form discrete concepts from sensory data; (2) ground language (n-grams) to these concepts; (3) induce a grammar for the language being used to describe the perceptual world; and moreover to do all this incrementally, without storing all previous data. The learning is achieved in a loosely-supervised manner from raw linguistic and visual data. Moreover, the learnt model is transparent, rather than a black-box model and is thus open to human inspection. The visual data is collected using three 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 analysis performed on these robots demonstrates the effectiveness of the framework in learning visual concepts, language groundings and grammatical structure in these three online settings.

Keywords: Language and Vision, Language Acquisition, Language Grounding, Grammar Induction.

1. Introduction

We aim to provide a machine with the ability to incrementally 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 objects and activities in their environments.

1.1. The Language Acquisition Problem

Language acquisition is the process by which an agent acquires the knowledge needed to comprehend natural language, as well as to produce meaningful words and sentences to communicate with others. The quest to find solutions to this problem has raised many issues. In this work, we attempt to tackle some of these by addressing the problem of learning perceptual categories, how language elements map to these, and what grammatical structures are used to talk about the perceptual categories. This is achieved by building a system which uses loosely-supervised learning of a language's syntax and semantics from a corpus of videos and descriptions featuring robots performing various tasks. This requires incremental learning methods that can operate on the outputs of various sensing modalities, such as RGB and depth cameras, language, etc. The outcome of this learning process is a collection of concepts, such as objects, relations

*Corresponding author
Email address: a.g.cohn@leeds.ac.uk (Anthony G. Cohn)

and activities that occur in the robot's environment, as well as their mapping to/from natural language such that the robot can understand given commands and be able to interact with humans. We have built a system that implements this approach, and for convenience of reference below, we name this system OLAV (Online Language Acquisition from captioned Video). Figure 1 illustrates the main functionalities of OLAV.

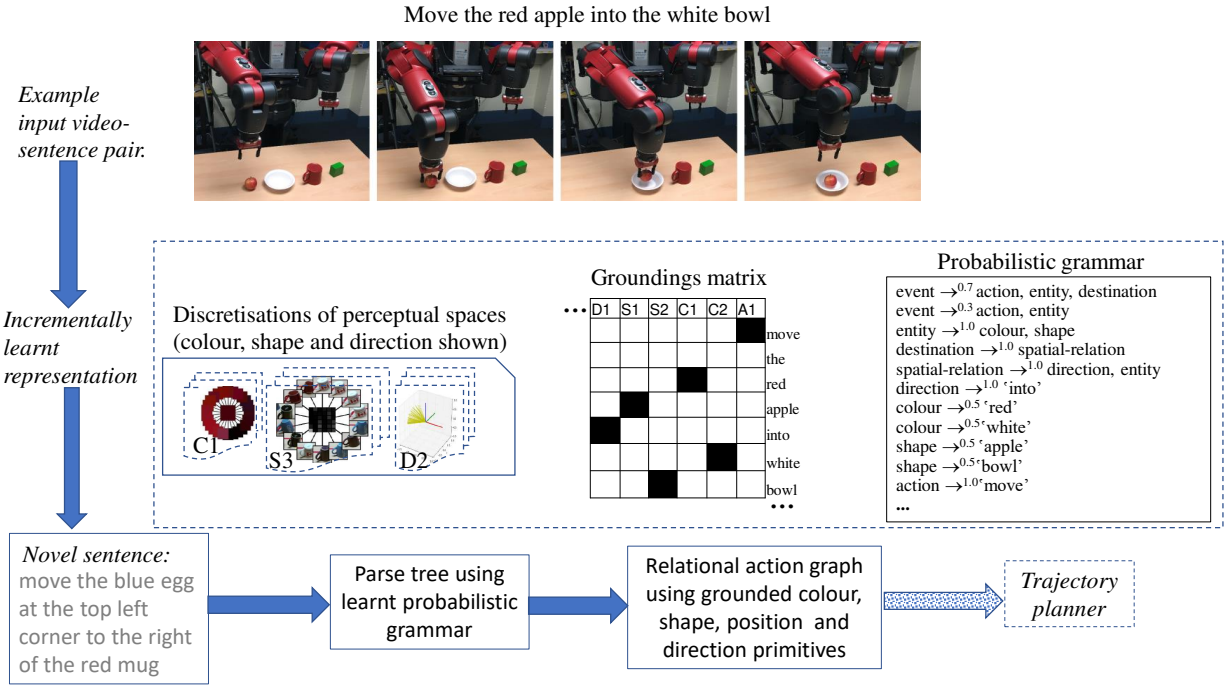


Figure 1: An overview of OLAV's functionality. (top) OLAV receives a video and a textual annotation describing the command being performed. (middle) OLAV incrementally learns conceptualisations of the input, including (left) discretisations of perceptual spaces such as colour, shape and direction), (centre) groundings of the n-grams found in the annotations, and (right) a probabilistic grammar induced from the annotations. (bottom) These learned representations can then be used to parse a novel sentence, and then transform it to an abstract relational graph-based perceptual representation. This could then be used to guide a trajectory planner, for an actual robot hand – this functionality is not currently part of OLAV– hence the shaded arrow to this final component. (This last functionality is not dissimilar from the functionality present in [1] where a low level manipulation trajectory is produced from a high level plan; but in OLAV the representation is entirely emergent from the examples over time.)

1.2. Structure of the Paper

In Section 2 we survey prior work in the field of language acquisition. In Section 3 we present the overall learning framework. In Section 4 we show how perceptual concepts are acquired incrementally, while in Section 5 we describe how language n-grams are grounded to these acquired perceptual concepts. In Section 6 we show how a probabilistic context free grammar can be acquired without any further human input in an online fashion. In Section 7 we validate the entire framework in three robotic settings and then draw the paper to a close and discuss possible future research directions in Section 8.

This paper represents an expansion and integration of portions of previous work published by the authors [2, 3, 4, 5]. It also includes material from the PhD thesis of the first author[6].

2. Related Work

One of the earliest computer systems capable of understanding natural language commands to perform tasks was SHRDLU [7], which was equipped with all the knowledge needed to understand linguistic com-

mands in a simulated robotic world. Since then, there have been many systems which learn at least some aspect of the knowledge that SHRDLU was pre-given, but to the best of our knowledge all these systems also exploit pre-given knowledge. A principal novelty of OLAV is that it makes very few assumptions compared to other work. Below, we first briefly survey some of this work, focusing on grounding language for robots, and then in Table 1 we summarise the difference in assumptions made by OLAV compared to other systems.

The field of language acquisition contains within it a large number of research areas, including: (i) The grounding of language to vision (learning a semantic representation of language), and (ii) the grammar rules of language (learning a syntactic representation of language)[1]. These two areas, which are surveyed in Section 2.1 and Section 2.2 respectively), 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. Moreover, it has been argued [8] that in general, learning language without experience grounding the meaning of language severely hampers language understanding research.

2.1. Language grounding

Spranger and Beuls [9] note that *referential uncertainty* is often thought to be the most important aspect of word learning. Quine [10] describes referential uncertainty as the problem of how words get their meanings, which is a general question anyone faces when trying to learn a new language. The space of possible meanings for a new word is unbounded. A word can be used to refer to anything from a physical object (e.g. book), a feeling (e.g. love), etc. The language grounding problem can be thought of as a subset of the referential uncertainty problem, where the space of possible meanings is limited to concrete observable ones, such as learning about physical objects.

Siskind [11] focused on understanding how children learn their native language, and how their language is mapped to their visual representation of the world. In many ways, his research is the closest in spirit to our present proposal. Later work from Siskind's lab includes Yu et al [12] which uses a pre-given grammar but otherwise presents a comprehensive framework for language grounding.

Needham *et al.* [13] 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 learned from it. The language in the audio was limited to one word at a time.

Whereas most work considers only the grounding of objects, and their properties, and sometimes actions and spatial relations too, there has been some work on other aspects of language such as comparatives, quantifiers and proportions, for example Pezelle et al [14], and Rahgooy et al [15]. Moreover, some research has addressed the problem of learning the grounding of specific action words or verbs (e.g. [12, 16, 17]).

2.1.1. Language acquisition in robots

An early work to try and learn language grounding for robotics applications is Roy *et al.* [18]: their system was capable of learning objects' names, which used mutual information criteria from recorded images and audio data. Since then, there has been a great deal of further such work.

One work sharing some similarities with OLAV is that of Sinapov et al [19], who present a system in which a robot learns various object properties and binary relations by using programmed robot exploration and supervised labelling. A similar approach is taken in Thomason et al [20] where the robot learns groundings via an *I spy* game, which provides input for a supervised SVM-based model. This approach was used in subsequent work such as Thomason et al [21] who use word embeddings to help ground novel words[2].

Another approach which has a number of similarities toOLAV is that of Yu et al [16]; they assume a fixed lexicon and a pre-existing parse (and thus do not attempt grammar induction) but present method which exploits the compositionality of events and language which is able to learn meanings of words and some n-grams covering objects, spatial prepositions and some actions. Whereas most work aims to learn

[1] This is not to suggest that syntax and semantics are completely separate; indeed it is a basic purpose of grammar to help map from form to meaning, and meaning may influence a speaker's choice of grammatical form. But it is still useful to distinguish between these two aspects of language in our review.

[2] An interesting aspect of this work is that non visual predicates such as heavy, or rattles, are learnt as well as visual ones.

language grounding from real world data, others consider learning from synthetic data. A disadvantage of this is that there can then be discrepancies or misalignment between the learned model and real world data. One attempt to bridge this gap is Can et al. [22] who investigate how to reason about the grounding of an instruction and a robot's world representation. Researchers have also used web-available descriptions and images to teach robots how to perform different tasks. For example, Beetz *et al.* [23] implemented a system that used descriptions from a *wikihow* website to teach a robot to make pancakes.

A recent survey of language use and learning in robotics is Tellex *et al* [24]. Two classifications of work in this area are offered: (i) by technical approach and (ii) by problem addressed. Learning is one of the three technical approaches covered (the others being logic-based methods), and methods that focus on HRI. They list ten common datasets[3] used for language grounding in robotics; of these, most only contain static images and do not support action learning; of the others, they are largely focused on navigation rather than the table top manipulation worlds we focus on in this paper, though the SLU dataset[4] does include manipulation actions for a forklift truck. They note that many of the learning approaches use already defined languages, rather than learning the language as we do here with OLAV. They also distinguish between systems that learn a mapping to prespecified visual concepts, and those that learn new visual concepts, and that the latter is more challenging as a learning problem. They also note the increasing amount of work based on deep learning. Some of this treats the problem as a language translation problem (from language to a sequence of actions or goals); in general the set of tokens to ground to is prespecified (unlike in our work). Common to all these deep learning based approaches is that they are not online, i.e. do not support incremental learning, require large training datasets (and hence are not particularly suitable for robotic applications where acquiring such data can be problematic), and are black-box methods. Like them, we do not delve further into this area, as the methods and goals of such work differ markedly from ours.

2.2. Grammar induction

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.

Most researchers have tackled the grammar induction problem in a *supervised manner* to enable their robots to understand natural language commands. The learning is achieved by providing the robot with input sentences along with their manually annotated grammar trees, be it syntactic or semantic trees. For example, the works of Dukes [26] and Wang *et al.* [27] used a supervised approach to learn how to parse natural language commands for manipulation tasks into a formal representation that a robot can understand. Another example of grounded language acquisition and grammar induction is that of Ross et al [28] who used paired sentences and videos to help train a semantic parser, which already has knowledge of Part-of-Speech (POS) syntactic types for its lexicon. In Matuszek et al [29] it is shown how to jointly learn language and perception models, but the approach is demonstrated for object property learning, but not for actions, or spatial relations. A follow on work [30] built on this system and demonstrated its application to other languages (Spanish and Hindi).

There has also been a small amount of work on *unsupervised learning* techniques to tackle the grammar induction problem from unlabelled sentences. For example, Chen and Mooney [31] 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().* While unsupervised grammar induction techniques enable learning from unlabelled data, their performance is usually significantly worse than those of the supervised techniques.

[3] A new challenge dataset for which a state-of-the-art neural baseline performs poorly is presented in Shridhar et al [25].
[4] http://people.csail.mit.edu/stefie10/slu. Last accessed 24 Aug. 2020.

2.3. Summary

There has not been space here to have provided a comprehensive summary of the field, but we have aimed to cover representative approaches and methods. In all the works described above, and indeed all other research we are aware of, the language acquisition problem was simplified by using at least one of the following 10 assumptions – Table 1 shows which systems made which assumptions.

1. The presence of 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 *the* in *pick up the ball*.
2. Only individual words are to be grounded rather than n-grams composing a phrase to be grounded.
3. Predefined 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.
4. A set of predefined atomic actions, spatial relations, and/or object classes that are used as the space of possible meanings of language, rather than learning from raw vision data.
5. A teacher that supervised the learning of language grounding and provides constant feedback to correct any mistakes.
6. A very limited number of objects in the world, sometimes just one or two, greatly simplifying the grounding problem.
7. A 1:1 correspondence between words and semantic concepts.
8. Only certain aspects of language are to be learned (e.g. only object properties, but not actions or spatial relations).
9. The learning is non-incremental – i.e. all training data is supplied in batch mode, rather than incrementally with the model being updated in an online manner.
10. Background world knowledge is exploited in the learning process.

Exploiting one or more of these assumptions (and in most cases more than one) simplifies the learning of language in each system discussed above and may have enabled a focus on learning more complex concepts, such as making pancakes. However, in this paper, we focus on natural language acquisition itself, and present novel techniques capable of acquiring semantic meanings of words and phrases from unlabelled linguistic and vision data, all in an online fashion, (without storing all previous instances). Our contribution is best regarded not as an improvement on the state of the art surveyed above, but rather a radical alternative methodology to address the language grounding and acquisition task since there is no previous work which requires so little supervision and so little in the way of predefined knowledge and assumptions.

3. Learning Framework

We aim to answer the following two questions: (*i*) can a robot bootstrap its knowledge in language and vision? (*ii*) can a robot ground language to concepts in vision? In this paper we present OLAV, a novel approach capable of acquiring symbolic knowledge of both language and vision concurrently, incrementally and in a loosely supervised manner. Furthermore, we also regard it as desirable that the outcome of the learning process in OLAV is representable in a human understandable form.

The learning is accomplished using a show-and-tell procedure, by presenting OLAV with snippets of data consisting of videos and text. The learning videos come from recording volunteers controlling a robot to perform a variety of table top tasks. The videos were subsequently annotated with appropriate linguistic descriptions as shown at the top of Figure 2. The recorded videos and descriptions are used as input data to OLAV to learn three key components: (*i*) the visual representation of the world; (*ii*) the groundings of words and phrases to the learned visual representation; and (*iii*) grammar rules which generate the language. To the best of our knowledge, this is the first system that learns all these three components, and with only minimal prior knowledge of any of them.

Figure 2 illustrates the main steps which OLAV performs in order to incrementally update its representations with a new video accompanied by an annotation while Figure 3(b) shows all the components of OLAV. The framework is applied on every video-sentence input, and the cumulative knowledge in language and

Ref. \ Assumption	1: Stop words provided?	2: Only 1-grams?	3: Grammar provided?	4: Predef. perceptual concepts?	5: Strong supervision?	6: Limited number of objects?	7: 1:1 groundings?	8: Which groundings learned?	9: Batch mode?	10: Background knowledge?
Siskind (1996) [11]	N	Y	N	Y (no vision component)	N	N	*:*	O,P,A	N	N
Needham et al (2005) [13]	n/a	Y	n/a	N	N	Y	1:1	O,A,R	N	Mode declarations
Yu et al (2018) [12]	Y	N	Y	Learnt by supervision	N	N	1:1	O,P,R,A	Y	N
Yu et al (2015) [16]	Y	Only as defined by the grammar	Y	Y, or learnt by supervision	N	Y	1:1	O,P,R,A	Y	Defns of actions and relns.
Roy et al(1999) [18]	N	N	n/a	N	N	Y	1:1	O	Y	N
Sinapov et al(2014) [19]	n/a	N	n/a	N	Y	Y	1:1	O,R,P	N	N
Thomason et al(2016) [20]	Y	Y	n/a	N	N	Y	1:1	O,P	N	N
Can et al (2019) [22]	Y	Only as defined by the grammar	Learnt by supervision	Y	N	Y	1:1	O,P,R,A	N	N
Nevens et al (2020) [32]	n/a	N	N	N	Tutor	N	*:*	O,P	N	N
Lauria et al(2002) [33]	n/a	Y	Y	Y	Y	Y	1:1	O,A	N	Y
Huang et al (2017) [34]	n/a	Only as defined by the grammar	Y	Y	Y	N	1:1	O,R,A	Y	N
Tellex et al (2011) [35]	Y	N	Y	N	N	N	*:1	O,R,A	y	N
Matuszek et al (2013) [36]	Y	N	Y	Y	N	N	1:1	O,P,R,A	Y	
Barrett et al (2018) [37]	Y	N	Y	Learnt by supervision	N	N	1:1	O,R	Y	N
Patki et al (2019) [38]	n/a	Y	Y	except objects	Y	N	Y	O	N	N
Roesler et al (2019) [39]	N	N	N	Y	N	Y	*:1	O,P,R,A	N	N
Thomason et al (2017) [40]	Y	Y	N	N	Active learning	Y	1:1	O,P	N	N
Steels et al (2000) [41]	n/a	Y	n/a	N	N	Y	1:1	O	N	Action models
Guadarrama et al (2013) [42]	Y	Y	Y	Learnt by supervision	Y	N	1:1	R	Y	N
She et al (2014,2017) [43, 44]	Y	Only as defined by the grammar	Y	N	Y	n/a	1:1	O,A	N	Y
Spranger et al (2015) [45]	N	Y	N	N	N	Y	1:1	O,R	N	N
Thomason et al (2020) [46]	Y	Y	CCG lexicon & categories	Y	Y	N	*:1	O,P	N	ontology
Matuszek et al (2012) [29]	Y	Y	Categories provided for all but object (attributes)	N	Supervised initialisation. Pointing to objects	N	*:1	O,P	N	N
Roy et al (2002) [47]	N	N	N	N	N	N	1:1	O,P,R	N	N
OLAV	N	N	N	N	N	N	*:*	O,P,R,A	N	N

Table 1: A table showing, for the works most closely related to OLAV discussed above, which of the assumptions 1-10 are made. OLAV makes none of these assumptions. Note that we only include models with a learning component and where the main learning mechanism is non-neural in this comparison, i.e. methods which learn a transparent model. In the *1:1 groundings* column, an entry of 1:1 means both of these must be unique; 1:* means that an n-gram can refer to multiple concepts, and *:1 that multiple n-grams can refer to a single concept, and *:* that neither need be unique. In the *Which groundings learned* column, the letters have the following meanings: O=Objects, A=Actions, R=Spatial Relations, P=Object Properties. In some cases the source paper did not make the answer explicit, so the entry was inferred to the best of our ability from our reading. Note that there are other distinguishing aspects of some systems which are not tabulated here. For example, some systems ground places descriptions – here we have treated these as objects (O). In the table we include some work for which there was not sufficient space to explicitly discuss it above, but which is still related to the goals of our research.

vision is updated *incrementally* with each processed video-sentence pair. It should be noted however, that while learning visual concepts starts from the very first video-sentence pair, it may take a number of these before there is sufficient evidence for grounding hypothesis generation to commence, and then for grammar induction to start.

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 the spatial relation *left of*. As the robot learns the visual concepts, the

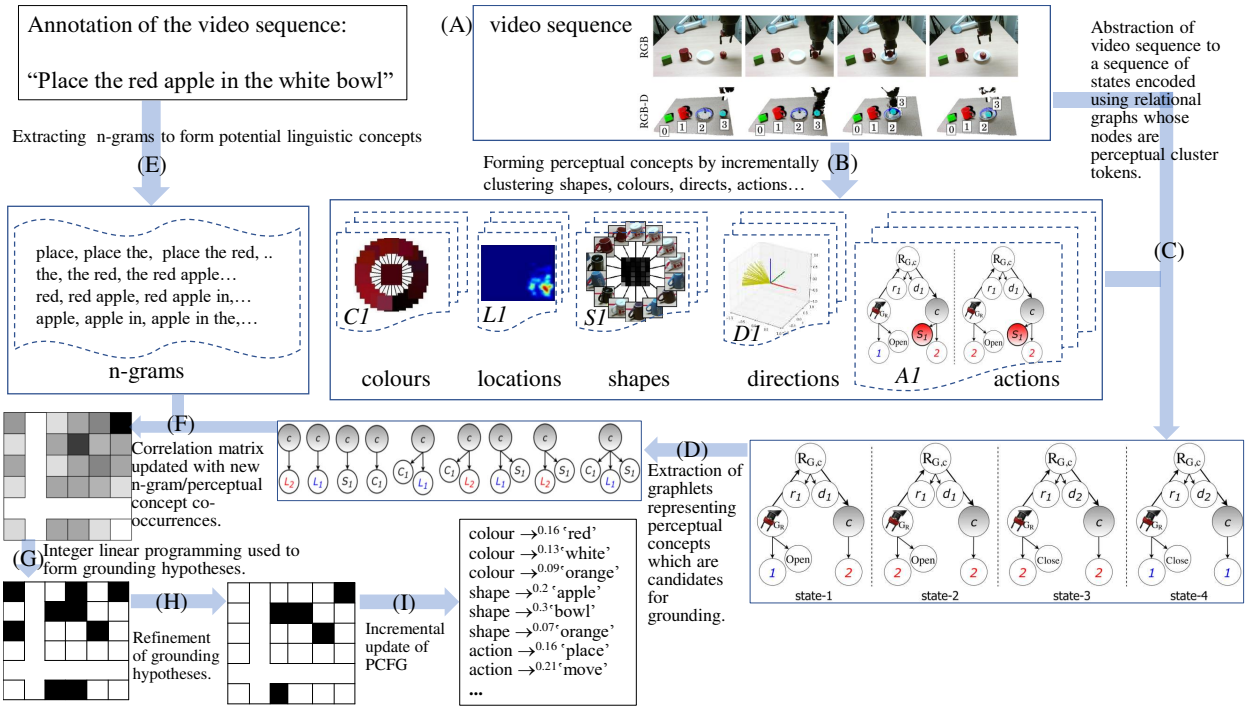


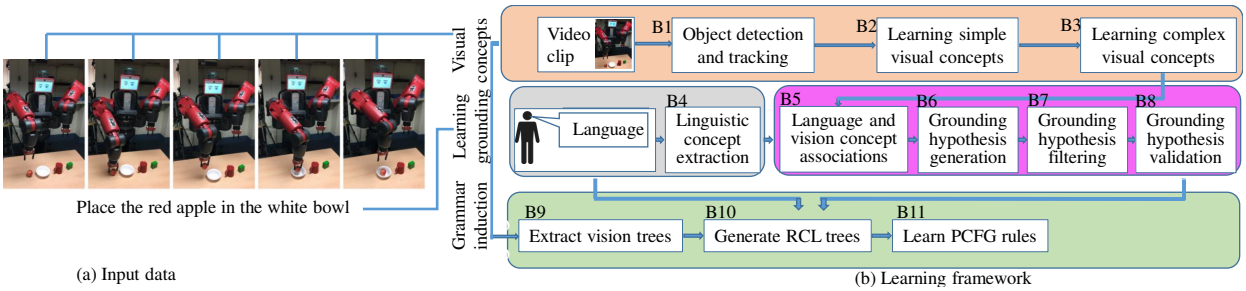
Figure 2: The process by which OLAV incrementally updates its representations to learn perceptual categories, groundings of n-grams to these, and a Probabilistic Context Free Grammar (PCFG). **(A)** Objects are recognised in the RBG-D frames and subsequently tracked. **(B)** Perceptual categories are formed from these objects by incrementally clustering the properties (such as colours, locations and shapes) and relationships (such as distance and direction) between them; one example (labelled *C1*, *L1*, *S1*, *D1*, *A1*) of each kind of depicted category is illustrated in the front 'tile'. **(C)** A video sequence can now be abstracted as a sequence of states each of which is represented as a relational graph whose nodes are the perceptual categories formed in (B). **(D)** Graphlets (i.e. subgraphs of the state graphs) are extracted as these are candidates for grounding. **(E)** Potential linguistic concepts are extracted from the annotation by forming n-grams. **(F)** A correlation matrix which represents the frequency of co-occurence of n-grams (rows) and perceptual categories (columns) is updated. **(G)** Integer Linear Programming is used to decide which correlations should form grounding hypotheses. **(H)** The hypotheses are filtered to eliminate inconsistent possibilities. **(I)** New rules can be added to the current PCFG, and probabilities updated for existing rules. Note that it is possible for a single linguistic concept (e.g. 'orange' to refer to two different perceptual concepts. This process is repeated for each new annotated video.

natural language descriptions are then 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.

We summarise here the inputs and outputs of each of the boxes (labelled B1-B11) in Figure 3. We also give pointers to the sections where each of the boxes are explained in more detail below – the overview here is necessarily very brief, and intended principally to catalogue the input-output relationships between the different components of the architecture. It also serves as a road map to the rest of the presentation of the architecture below. We mark with "(*)", those aspects we regards as being particularly novel.

B1 Object detection and tracking. INPUT: Video represented as a sequence of RBG-D frames. OUTPUT: tracked objects located in time and space. See Section 4.2

B2 Learning simple visual concepts. INPUT: tracked objects from (B1) and the current clusters for each space learned from previous videos. OUTPUT: Updated sclusters of each simple concept perceptual space (e.g. properties of objects such as colour, shape, distance, and relationships between objects, such as distance and direction). Each perceptual space is a continuous space, but is discretised as a result of this step, with new concepts being added as further videos are processed, and existing concepts adjusted. Incremental Gaussian Mixture Models (IGMMs) are used to perform this clustering incrementally. See Section 4.4. (*)



(a) Input data

(b) Learning framework

Figure 3: (a) Examples of input video clips annotated with natural language commands. (b) Our learning framework, consisting of three main components: (i) learning of visual concepts (the orange block), (ii) natural language grounding (the purple block) and (iii) grammar induction (the green block). The framework is applied to every video-sentence input pair, which can then be discarded after processing. (Best viewed in colour.)

B3 Learning of complex visual concepts. INPUT: tracked objects from the current video (B1) and the perceptual categories from (B2) for each object (e.g. colour and shape) and each pair of objects (e.g. distance and relative direction). OUTPUT: a sequence of relational graphs; each graph represents a qualitative temporal state and the objects involved and their discretised perceptual categories; if the sequence is not novel, the action space is not updated. See Section 4.5.

B4 Linguistic concept extraction. INPUT: the annotation for the current video and the list of n-grams from all previous videos. OUTPUT: all the n-grams from the annotation (with $n \leq 3$) are added to the overall list of n-grams. See Section 5.2.

B5 Language and vision concept associations. INPUT: (i) A correlation matrix, K, computed from all *previous* video-language pairs which has recorded the frequency of co-occurence of each n-gram/perceptual category pair; (ii) the list of n-grams in the *current* video from (B4); (iii) the list of perceptual categories in the current video from (B2) and (B3). OUTPUT: An updated correlation matrix K′. See Section 5.3.

B6 Grounding hypothesis generation. INPUT: the correlation matrix, K, from (B5). OUTPUT: Integer Linear Programming is used to turn each square in the correlation matrix, K, into either a 1 (representing the hypothesis that that row (n-gram) can mean that column (perceptual concept), or a 0, representing that the n-gram is not associated with that perceptual concept; we call this map, $\mathcal{A}$; it is represented visually as a 0-1 (white/black) matrix. See Section 5.4. (*)

B7 Grounding hypothesis filtering. INPUT: A map, $\mathcal{A}$, (i.e. the output of (B6). OUTPUT: A modified map, $\mathcal{A}'$, possibly with some entries which were 1 now 0. See Section 5.5.

B8 Grounding hypothesis validation. INPUT: A map, $\mathcal{A}$, (B7); also the sequence of relational graphs for the current video from (B3) OUTPUT: A modified map, $\mathcal{A}'$, possibly with some entries which were 1 now 0. See Section 5.6. (*)

B9 Extract vision trees. INPUT: The sequence of relational graphs for the current video from (B3). OUTPUT: A *vision tree* for the current video. A *vision tree* which is a tree which contains the three main components of a command to the robot (the action, the entity and the destination) – see Figure 14.

B10 Generate RCL trees. INPUT: the vision tree from (B9); the text annotation of the current video. OUTPUT: An RCL (Robot Control Language) tree which matches the input vision tree – see Section 6.2 (*).

B11 Learn PCFG rules. INPUT: The existing PCFG so far (an empty grammar initially); the RCL tree from (B10). OUTPUT: a modified PCFG. See Section 6.4.1.

3.1. Assumptions and Limitations

In our learning framework we make 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. The main assumption we make is on *loosely-supervised* learning, while some further assumptions are discussed in the remainder of this paper. We use the term *loosely-supervised* to describe the learning process that requires the videos and sentences to be temporally aligned beforehand i.e. for each video there is specific text that is associated with it, in our case which describes the action in the video; the question as to how continuous activity is segmented into discrete video-text pairs is not addressed in this paper – but see the comments in further work, Section 8.4.1. We consider the learning in OLAV to be loosely-supervised rather than supervised or unsupervised: the videos in the datasets described below are all fairly short and depict a single action, or a very short sequence of connected actions, and the linguistic annotation describes precisely these action(s), with no distracting uncommented actions (though there may be objects in the scene not described). An unsupervised system would be able to learn from longer non-segmented videos and documents, or even learn from a constant stream of audio-video data, which remains an ambition for the future. A fully supervised system would require a specification of precisely which words correspond to which perceptual objects and actions, thus also pre-defining the parts of speech for words, such as those systems labelled "Y" in Table 1, column 4.

4. 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, a colour represented as a cluster of values in the HSL colour space is considered a visual concept. We present in the following sections the robots, sensors and feature spaces used, along with the unsupervised methods employed to generate such concepts.

4.1. Robots and Sensors

Three different robots are used to validate our learning approach: (*i*) A Baxter robot from Rethink-Robotics (named LUCAS) that has two arms and two fingered grippers; (*ii*) A custom made mobile manipulator by Sinapov *et al.* [48] that uses a 6-DOF Kinova Mico arm[5]; and (*iii*) A simulated 3-DOF robotic arm with a two fingered gripper in a chess-board simulation environment that we developed and presented in Alomari *et al.* [2]. The three robots are shown in Figure 8 which appears in Section 5.1. The robots are equipped with at least one sensor that allows mapping of the environment, such as a Kinect2[6] that allow collecting RGB video streams in addition to depth point clouds.

4.2. Low-Level Processing of Input Data

The robots are used to collect short video clips of the environment, where each video contains one action performed, e.g. a robot moving an object, a robot dropping an object, etc. Each recorded video clip is processed to detect and track objects in the scene. For each video clip, OLAV encodes a number of visual representations by initially detecting objects in the video using a table-top object detector [49]. Once an object is detected in a video clip, the location of this object is tracked across all remaining frames using a particle filter tracker presented by Klank *et al.* [50].

4.3. Concept Extraction

Concepts are learned automatically by clustering the low-level sensory input of each of the robot's sensor modalities after an appropriate encoding. This clustering operation results in a collection of classes that are candidate concepts within each feature space. Because OLAV has no prior knowledge of the structure of the sensor feature spaces, it uses probabilistic modelling techniques for each feature space independently to elicit meaningful classes that are supported by the observed data.

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

[5]Robotnik, http://www.robotnik.eu/robotics-arms/kinova-mico-arm/ – last accessed in November 2018.
[6]Microsoft, http://www.microsoft.com/en-us/kinectforwindows/meetkinect/features.aspx – last accessed in October 2017.

using geometrical and textural cues. On the other hand, (*ii*) *complex concepts*: ones that manifest over longer sequences of observations. For instance, temporally-extended robot actions are examples of complex concepts. For these, a more elaborate encoding and more sophisticated clustering mechanism is needed. We discuss both simple and complex concepts below.

4.4. Simple Concepts

Our simple concept acquisition is demonstrated by extracting two types of concepts from raw data. The two types are: (1) object properties, and (2) spatial relations.

4.4.1. Object properties

For each detected object, OLAV aims to learn about its properties (shape, colour, and location). This is achieved by clustering the values in these continuous feature spaces into a number of 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.

To learn shapes, OLAV uses the fast point feature histogram (FPFH) representation [51]. AFPFH is a multi-dimensional histogram of features which describe the local geometry around a point p in a 3D point cloud. The FPFH values from each scene are clustered to generate visual concepts using Gaussian mixture models and a Bayesian Information Criterion (BIC) as shown in Figure 4. The resulting Gaussian components are used as concepts to represent unique shapes in the environment. Similarly, for colours and locations, the values for these two are extracted for every object and clustered. Colours are measured in HSL space, while locations are measured by the centre location (x, y, z) of each object.

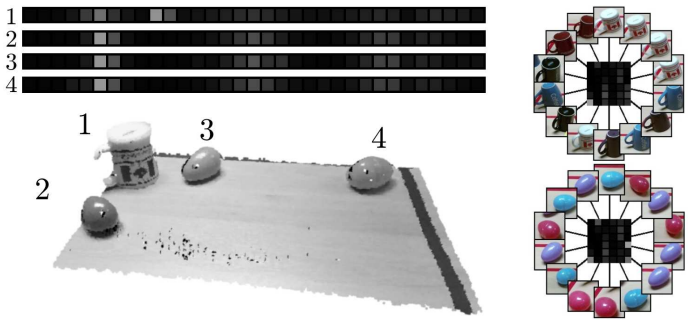


Figure 4: Left: Examples of Fast Point Feature Histograms for four objects in a point cloud. Right: Examples of two different object clusters with the averaged (mean) values of each of the 33 bins of FPFH shown in the centre of each cluster.

4.4.2. Spatial relations

For every pair of detected objects in a scene, two pair-wise spatial relations are computed from their centroids: Euclidean distance, and relative direction (azimuth and altitude angles), where $distance : object \times object \rightarrow R$ and $direction : object \times object \rightarrow [0, 360) \times [0, 360)$. OLAV computes the azimuth and altitude angles from the point of view of the observer (the robot) as it assumed that the individual objects do not have a main or principal axis or a front face to compute such angles from. Thus the learning of spatial relations such as left and right is limited to egocentric rather than allocentric relations at present. Also, learning is presently limited to simple pair-wise directions and distances as opposed to learning comparatives or superlatives too, such as *further right, rightmost*, etc. as learning these concepts requires a more complex representation and inference mechanism than is present in the current version of OLAV.

4.5. Complex Concepts

We define complex concepts as ones that manifest over longer sequences of observations. In this paper we restrict OLAV's task to learning about robot activities; see [52] for how we built a system that shares much of its architecture with OLAV and learns about human activities.

To learn about complex concepts, i.e. robot-action concepts, the robot is controlled by volunteers to demonstrate how to perform different table-top tasks. The learning happens in the following manner: we think of a command for the robot such as 'move the blue egg onto the mug' and record this as textual annotation; then a volunteer drives the robot arm using a joystick to perform this action while the robot records the environment using an RGB-D sensor. Using the recorded videos, the robot learns about the different actions using three processes: first, encoding the visual world into a number of predicates; second, abstracting the changes in the visual world using *spatio-temporal graphs*; third, mining these graphs to obtain sub-graphs (or graphlets) that are used as representation for the robot actions.

Visual world encoding: The objects and relations that are involved in each action are represented using a number of predicates. Each video clip is processed to extract the unique spatial and object related concepts. The representation is made using a collection of predicates of the form: $object\text{-}concept(object)$ for object properties; and $spatial\text{-}concept(object_1, object_2)$ 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 $colour_1$. Each object and relation is represented using these internal symbols, which is decided using the Mahalanobis distance [53]; the observation 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, and the distribution is an extracted Gaussian component that represents a simple concept. This process is repeated for every 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.

Spatio-temporal graphs: Spatio-temporal graphs are Directed Acyclic Graphs (DAGs) [54] comprising three layers. These graphs have been used in the literature to model human activities as presented by Sridhar *et al.* [55, 56], Gatsoulis *et al.* [57] and Duckworth *et al.* [58]; an example of these graphs is shown in Figure 5. 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 [59].

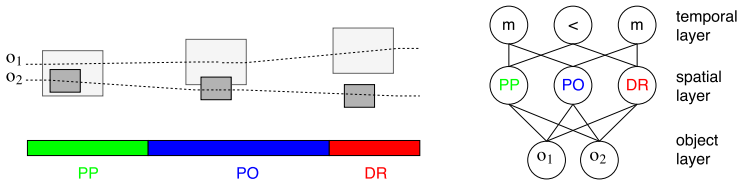


Figure 5: Spatio-temporal DAGs. (top-left): two objects o_1 and o_2 moving away from each other with every frame. (bottom-left): the QSRs between the two objects at every frame using RCC5 [60] with relations Proper-Part (PP), Partially-Overlapping (PO) and Discrete-Regions (DR). (right): the spatio temporal DAG for this scene, the temporal relations between the QSRs using Allen's Interval Algebra. The temporal relations are meets (m) and before (<).

Our spatio-temporal graphs differ in three ways: first, more properties and relations are encoded into each layer to allow for more complex representation of the world; second, our graphs use extracted concepts that OLAV learns by clustering the video clips as opposed to predefined ones such as the RCC5 in Figure 5; third, temporal relations are abstracted into a sequence of states holding over a maximal interval, where each state represents a constant qualitative configuration of the visual world; rather than using Allen's relations. An example demonstrating OLAV's graph representation is shown in Figure 6.

Extracting concepts from spatio-temporal DAGs (graphlets) The principle OLAV uses for learning the mapping between language and vision 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, learning might perhaps be performed on all possible sub-graph structures, but this remains an ambition for the future. In the research reported here, learning is steered 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

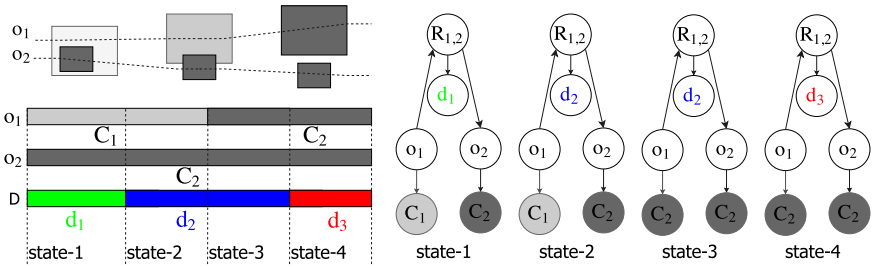


Figure 6: Extended DAG representation for two moving objects with changing properties: (top-left) two objects o_1 and o_2 moving away from each other, and object o_1 is changing its colour from *white* to *black* (bottom-left) this video has two colour concepts (C_1=*white*, and C_2=*black*), and three 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 whenever a change occurs in any of them, a sequence of states is generated.

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* used to connecting different graphlets together, thus, enabling the robot to reconstruct a complete spatio-temporal DAG. This ability will be used later to enable learning of word meanings, and grammar rules.

4.6. Continual Learning of Visual Concepts

In our incremental learning process, the robot is introduced to new visual concepts over time, e.g. new colours, spatial relations, etc. We utilise unsupervised incremental modelling techniques to update the learning of simple visual concepts. Note, we do not cover the incremental updating of complex concepts, i.e. robot activities, since, in our current representation when an activity is identified in a video it is either identical to a previous one, or not – we leave it for future work to consider how variants of robot activities could be learned; some work in this direction includes [56, 61]. OLAV uses the Incremental Gaussian Mixture Model (IGMM) technique presented by Song and Wang [62]. An IGMM is used to create Gaussian models to represent newly observed concepts and update previously learned ones, thus allowing OLAV to link varied observations across videos. This variation can happen owing to a number of reasons such as different lighting conditions when the videos were recorded, observing concepts from different view points, occlusions, etc. The IGMM technique is illustrated in Figure 7. The use of IGMMs to model concepts allows OLAV to efficiently update its models using a single pass over the data, optimising both storage and computation complexity, making it ideal for incremental learning.

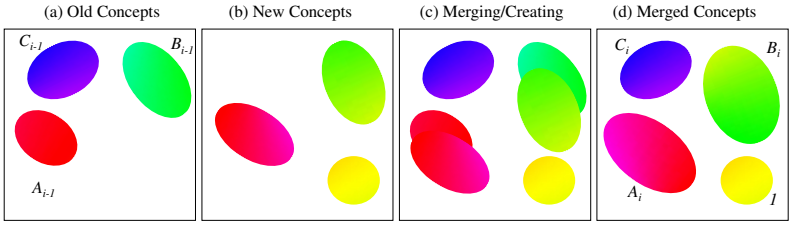


Figure 7: IGMM operating over Gaussian components in a hue-saturation colour space. (a) There are three colour concepts resulting for all video clips to date (*red, green* and *blue*, with frequencies A_{i-1}, B_{i-1}, and C_{i-1} respectively, assuming this is the i^{th} video that is now being processed. (b) In this i^{th} video, there are three clusters observed. (c) These are then merged with the previous concepts. Finally (d) shows the new, merged clusters, with their updated frequencies A_i, B_i, and C_i respectively. The new concept *yellow*, with an initial frequency of 1, is also shown in (d). (Best viewed in colour.)

5. Language Grounding

The robot has to learn the meaning of words without being able to ask direct questions 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, the space of possible meanings is limited to concepts that can be measured when a word is mentioned. More precisely, it is limited to 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. Next, we discuss the Language Grounding framework.

5.1. Grounding Framework

The grounding is achieved by learning a correspondence of words and phrases to some of the learned visual concepts in each scene. Ideally the robot would record the sound of speech and learn how components of this relate to the learned perceptual categories, but this remains an ambition for the future. At present, we collect multiple 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 8 from each of the recorded datasets used in this work.

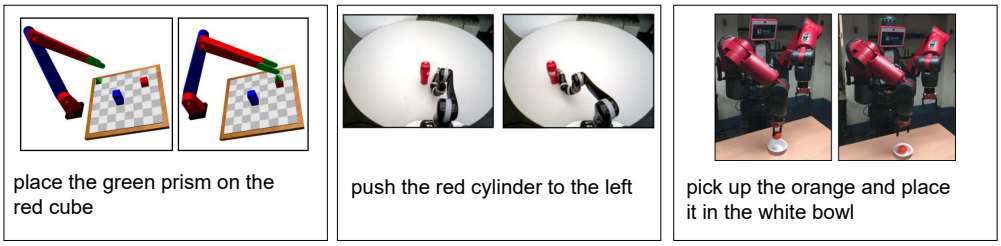


Figure 8: Examples of natural language descriptions collected for three different datasets. Each video clip is annotated with multiple sentences.

The aim is to learn the meaning of key words in the sentences by grounding them to visual concepts that represent their meaning. For example, OLAV should learn that the grounding of the word 'bowl' shown in Figure 8 (right) is the visual concept (i.e. Gaussian model) in the shape feature space (i.e. FPFH space) that represents how a bowl looks. The language grounding is achieved by following these five steps: 1) building a *language-vision* 2-D correlation matrix $\mathbf{K}(\delta, v)$ that measures the probability of associating linguistic (δ) and visual (v) concepts together; 2) using Integer Linear Programming to extract hypotheses from $\mathbf{K}(\delta, v)$ to create a map ($\mathcal{A}$) from language concepts to concepts in vision. 3) filtering the generated hypotheses using case analysis; 4) validating the remaining hypotheses through graph matching techniques and learn the correct grounding from language to vision; and, finally 5) updating the probability $\Phi(\delta, v)$ for each n-gram δ and visual concept v when they are matched. These steps are described in more detail in the following sections.

5.2. Preparing the linguistic and visual concepts for language grounding

For the vision domain, the input video clip is processed to generate a set of visual concepts as described in Section 4. The set of all learned visual concepts from all feature spaces are accumulated together into a single list $\mathcal{V} = \{v_1, \ldots, v_{|\mathcal{V}|}\}$, where v_i is a visual concept, e.g. a colour, a relation, a robot action, etc. I.e. the list $\mathcal{V}$ holds the cumulative knowledge the robot has gained about the visual world and is updated with every new video clip.

For the language 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. OLAV then extracts all possible n-grams from a sentence with $n \leq M$ (in our experiments in Section 7, M=3). The use of n-grams allows the learning of multi-word descriptions such as

'pick up','bottom left corner', etc. The list of all unique n-grams across all input sentences are combined into a list $\mathcal{N} = \{\delta_1, \ldots, \delta_{|\mathcal{N}|}\}$, where δ_i is an n-gram. The list $\mathcal{N}$ is updated with every new input sentence.

The robot now has two lists $\mathcal{V}$ and $\mathcal{N}$ acting as an intermediate representation for both vision and language domains. This representation transforms knowledge from continuous spaces to bounded discrete ones, and allows for the mapping between language and vision as we describe in following section.

5.3. Language and Vision Concept Association

As others have before, we rely on repetitions between words and visual concepts to teach our robots the meaning of words in the vision domain. 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 in OLAV's representation and therefore it should not be grounded to anything[7].

To measure the consistency of repetitions between concepts in language and vision, OLAV follows the frequentist approach presented by Everitt and Skrondal [63]. It keeps 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 visual concepts $\mathcal{V}$, and the set of all observed unique n-grams $\mathcal{N}$, we define a *concepts correlation* matrix $\mathbf{K}$ of size $b \times u$ and with n-grams as rows and visual concepts as columns.

The values in the *concepts correlation* matrix $\mathbf{K}$ are computed using Equation 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.

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

The first part of Equation 1 computes the strength of associating an n-gram (δ) to a vision concept (v). 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 always appearing together, and is equal to 0 if they are never seen together in the same pair of inputs.

The second part of Equation 1 (the exponential component) represents the certainty, or the learning curve of concepts. The aim is to penalise the learning of 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 1 is equal to 1). The same applies for linguistic concepts observed for the first time. The learning curve component in Equation 1 is an exponentially decaying function towards a limiting value that acts as a mechanism to penalise the learning of such concepts. 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 1 to equal 1.

Once the new input video-sentence pair is processed, the elements of the *concepts correlation* matrix $\mathbf{K}$ are updated according to Equation 1. Each element in the matrix $\mathbf{K}(i, j)$ represents the strength of association between 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 ground natural language to vision.

5.4. Hypotheses Generation for Language Grounding

A particular challenge is that mapping between words and vision concepts is not always one-to-one. For example, visual concepts can be described with different words, e.g. a block shape can be referred to in the English language by *block, brick, slab, bar,* etc. Moreover, as already observed above, words may be

[7]Of course 'the' could indeed be visually significant, indicating that the following referring expression is expected to be unambiguous; we leave this however for future work.

homonyms and refer to different kinds of entity, e.g. 'orange' might be a shape or a colour. To learn the grounding of language to vision, OLAV searches 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. 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:

$$\max_{\mathcal{A}} \sum_{\mathcal{N}\times\mathcal{V}} \mathcal{A}(\delta, v)\, \mathbf{K}(\delta, v). \tag{2}$$

We maximise the objective function with the following constraints:

- $\sum_{\mathcal{N}\times\mathcal{V}} \mathcal{A}(\delta, v)/(|\mathcal{N}| * |\mathcal{V}|) < \epsilon$, 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 sensitivity analysis on ϵ is performed in Section 7.3.2.
- $\sum_{\mathcal{N}} \mathcal{A}(\delta, v) > 1$, $\forall v \in \mathcal{V}$, forcing the assignment of at least a single n-gram to each of the learned visual concepts. This helps to ensure that each visual concept gets at least one word to describe it. 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, auxiliary verbs, etc. Note that this constraint might create 'false-positives' of groundings, but these will be filtered out in the validation step below.

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 9(right), 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.

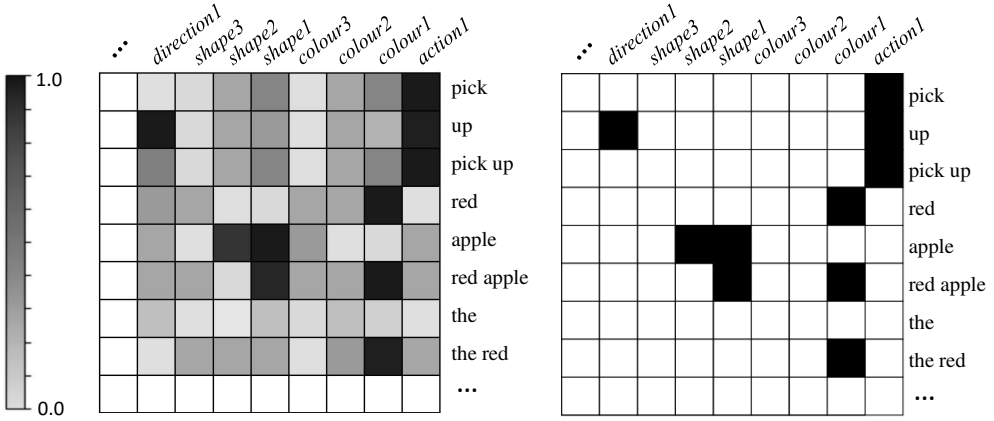


Figure 9: Grounding hypotheses generation. (left) The *concepts correlation* matrix $\mathbf{K}$. The value of each element varies from 0 to 1, where 0 means the two concepts were never observed together. (right) The target function $\mathcal{A}$ that results from solving the integer program for the matrix $\mathbf{K}$. Integer programming allows for multi-to-multi associations between n-grams and vision concepts. For example, 'up', 'apple' and 'red apple' are each mapped to two visual concepts.

5.5. 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, OLAV needs 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 the 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 (3 to 6) from which we can infer 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 3 all three n-grams are assigned to the same visual concept ($v_{ab} = v_a = v_b$), then OLAV can accept the hypothesis for the biggest n-gram $\mathcal{A}(\delta_{ab}, v_{ab}) = 1$ and filter out the smaller n-grams $\mathcal{A}(\delta_a, v_a) = \mathcal{A}(\delta_b, v_b) = 0$.

$$v_{ab} = v_a = v_b \quad \rightarrow \quad \mathcal{A}(\delta_{ab}, v_{ab}) \tag{3}$$
$$v_{ab} = v_a \neq v_b \quad \rightarrow \quad \mathcal{A}(\delta_a, v_a), \mathcal{A}(\delta_b, v_b) \tag{4}$$
$$v_{ab} = v_b \neq v_a \quad \rightarrow \quad \mathcal{A}(\delta_a, v_a), \mathcal{A}(\delta_b, v_b) \tag{5}$$
$$v_{ab} \neq v_a \neq v_b \quad \rightarrow \quad \mathcal{A}(\delta_{ab}, v_{ab}), \mathcal{A}(\delta_a, v_a), \mathcal{A}(\delta_b, v_b) \tag{6}$$

We now explain the reasoning for formulating each of the four rules. Rule (3) filters out the smaller incorrect n-grams, by allowing complex n-grams to subsume their constituent ones when the corresponding visual concepts are all the same. 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, 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 (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 10) the robot ends up with a number of candidate grounding hypotheses that require validation; details of the validation process are presented in the following subsection.

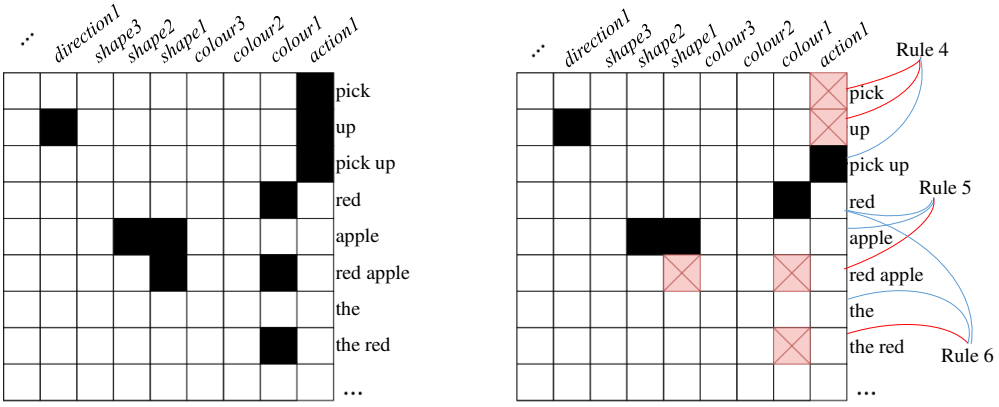


Figure 10: Filtering the grounding hypotheses. (left) The target function $\mathcal{A}$ obtained by solving the integer program. (right) The resultant target function $\mathcal{A}$ after filtering the grounding hypotheses using Rules 3 to 6.

5.6. Grounding Hypotheses Validation

Once grounding hypotheses have been filtered, OLAV attempts to validate them by using graph matching techniques. Imagine the scenario where the 1-gram 'apple' in the given input sentence 'pick up the apple' is grounded to two different visual concepts, one representing the shape *apple*, and the other representing something incorrect, e.g. the shape of a *mug* as shown in Figure 9. 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 (correlation alone would fail to learn the language grounding in this case, hence the need for validation). 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.

We refer to the graphs generated from connecting the graphlets together as *hypothesis graphs* as shown in Figure 9. Each *hypothesis graph* represents a different course of actions taken by the robot that reflects what it thinks the sentence means. For example, if the robot believes that 'apple' in the previous example ('pick up the apple') means the *apple shape*, it will pick up the *apple shape* in the hypothesis graph. On the other hand, if it assumes it means the *mug shape*, it will pick up the *mug* shape (Figure 11).

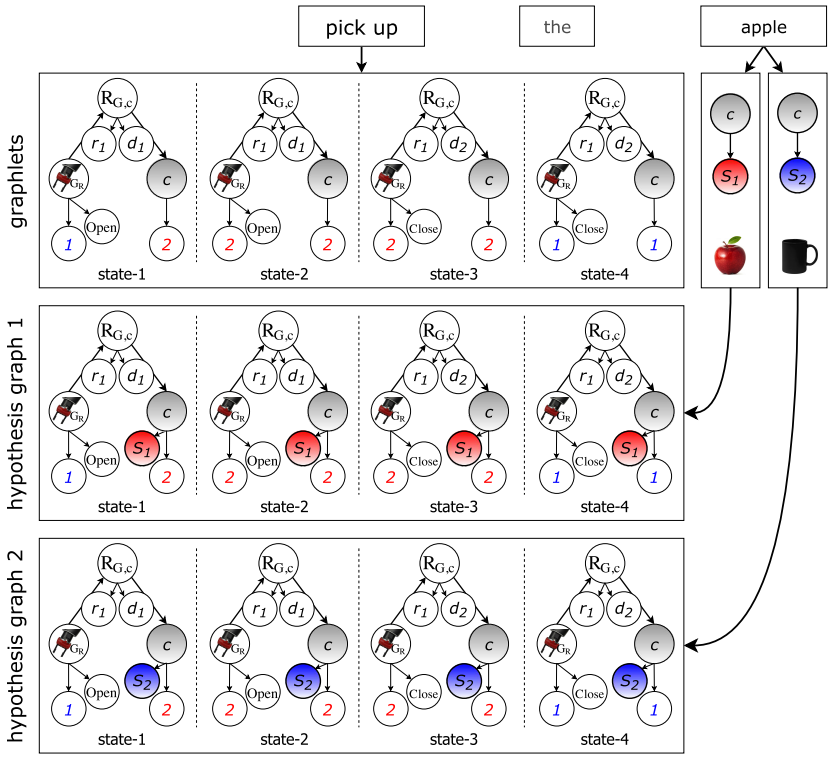


Figure 11: Generating *hypothesis graphs* from the sentence 'pick up the apple'. The word 'apple' has two hypotheses (S_1=*apple shape*, and S_2=*mug shape*). These vision concepts in their graphlet format are combined to generate two *hypothesis graphs*.

Each hypothesis graph generated from a sentence as described above is compared against its corresponding input video sequence. The idea is to look for a match between a hypothesis graph and the input video Spatio-Temporal graph. The matching is enabled by using an *induced sub-graph* matching technique presented by Howorka [64]. A hypothesis graph is said to match the input video if is an induced subgraph of the spatio-temporal DAG extracted from the input video. The procedure of how to probabilistically accumulate the knowledge of grounding hypotheses is described in the following subsection.

5.7. Learning Probabilities of Language Grounding, Φ

Once a *language grounding hypothesis* is validated as described in the previous section, the robot learns it through the grounding function Φ. This is achieved by updating the probability of this grounding hypothesis using Equation 7, where $\Phi : \mathcal{N} \times \mathcal{V} \rightarrow [0, 1]$; $P(v|\delta)$ is the conditional probability for a vision concept v given the n-gram δ, $\mu(\delta, v)$ counts the number of times the n-gram δ was validated with the vision concept v, and $\mu(\delta)$ is the total number of times the n-gram δ was validated with any vision concept. The probabilities in the grounding function Φ are updated incrementally as more grounding hypotheses are validated.

$$\Phi(\delta, v) = P(v|\delta) = \frac{\mu(\delta, v)}{\mu(\delta)} \tag{7}$$

5.7.1. Stop words

To simplify the learning of language grounding in robotics applications, it is common to use a stop word list to remove words such as 'the' and 'as' from all sentences. But, since OLAV learns from unlabelled data (i.e. avoiding human annotation including stop word lists), it learns 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 stop words (cf Jones [65]).

6. Grammar Induction

Unlike human acquisition of language which is largely unsupervised, nearly all computational approaches developed to learn about natural languages are supervised. In particular ones developed to learn the language structure (grammar), rely on human experts to provide training data labelled with grammar trees. An example is shown in Figure 12 for an annotated grammar tree from the Dukes [66] dataset. These trees are used to train a parser in a supervised manner to model the language structure.

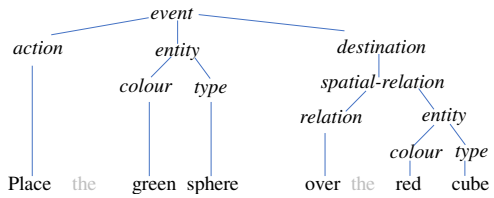


Figure 12: Example of an annotated grammar tree used as training data for supervised parsers.

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 is shown in Figure 13. We used the system presented by Ponvert *et al.* [67] to obtain this tree, by training it on the entire Dukes [66] dataset. This technique learns a language model via chunking the raw text into smaller parts that shows a repeated pattern.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. 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.

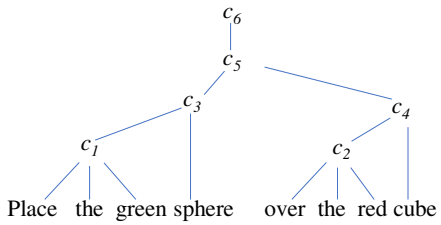


Figure 13: Example of an unsupervised grammar tree. The c_i represent an unknown label.

6.1. Learning Grammar from Language and Vision

In this section, 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. These trees are in fact *Robot Control Language (RCL) trees* [26]. As noted in section 2, the idea of learning grammar rules by mapping them to features in the vision domain has been introduced in the robotics literature before. In particular, Dominey and Boucher [68], Tellex *et al.* [35], Matuszek *et al.* [36] and Dukes [26] 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 RCL trees from sentences. The parsing of sentences into RCL trees enables robotic agents to execute linguistic commands that were not seen before in the training data. We aim to enable robots to understand natural language commands and descriptions without the use of labelled training data.

In the following subsections, RCL is described, along with how it is used to enable the learning of grammar rules.

6.2. Robot Control Language (RCL)

RCL is a tree semantic representation for natural language commands. Each sentence is represented as an RCL tree; an example is shown in Figure 12, 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 [26]. The RCL elements used in this work are presented in Table 2[8]. Each element represents one or more of the visual features, 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 [35] and Matuszek *et al.* [36].

RCL element	*Description*
event	Specification of a 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. *centre*, *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 2: The list of all RCL elements used in our experiments with OLAV. These RCL elements are designed to work in the context of robot manipulation.

The problem of parsing sentences into RCL trees has been formulated as a grammar induction problem. A parser is trained on commands and their human annotated RCL trees. The parser is then used to parse new commands into trees which the robot can understand and execute. The human annotation of RCL

[8]Of course, the world is much richer than is supposed by the set of elements in this table; these are the ones that the correspond to the feature spaces that are currently used by OLAV. Generalisation to other features and elements is considered briefly in the discussion of future work in Section 8.4.2.

trees is a labour-intensive task that hinders the learning from large datasets. OLAV does not suffer from this problem since it automatically generates a *vision tree* (Ω) from each input video clip. These vision trees 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 14. The v subscript in these elements refers to 'vision', to distinguish them from the equivalent RCL elements shown in Table 2. The $action_v$ element holds the internal symbol of the action that was performed in the video. The $entity_v$ element holds the *id* of the object that is manipulated by the robot in the video. 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.

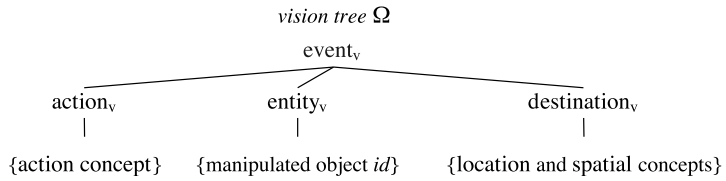


Figure 14: Vision tree Ω definition. The vision tree is an event tree, i.e. a tree with the $event_v$ element as its head. The $event_v$ element takes three children $\{action_v, entity_v, destination_v\}$.

6.3. Generation of RCL trees

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. For example, consider the video shown in Figure 15, paired with the command 'place the green sphere over the red cube'. OLAV extracts a vision tree Ω from this video clip with three elements shown in Figure 16. The $action_v$ element holds the internal symbol of the action graphlet extracted from this video clip (labelled with the internal symbol $action_1$). The same applies for the $entity_v$ element and the $destination_v$ element.

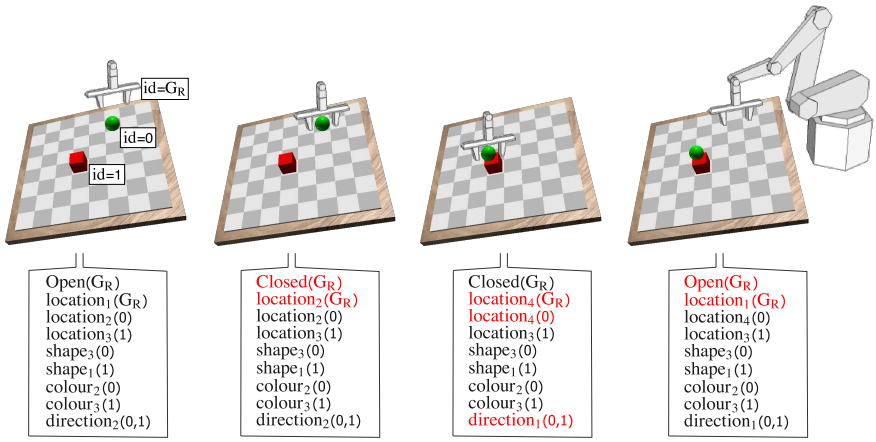


Figure 15: The input video clip that is generated for the command 'place the green sphere over the red cube' encoded with the learned visual concepts shown at different frames.

The problem of automatic generation of RCL trees in OLAV is formulated into a search problem as follows. For each input video-sentence pair, OLAV (*i*) extracts the vision tree Ω from the input video; (*ii*) generates the set of all possible RCL trees from the input sentence; (*iii*) searches for an RCL tree that matches the extracted vision tree Ω. OLAV aims to find the sentence structure that will result in a match with what happened in the input video. An RCL tree is said to match a vision tree if the values of their corresponding elements are equal. Given a match is found between these three elements in a language tree Ψ, OLAV uses this language tree to update the robot's grammatical knowledge.

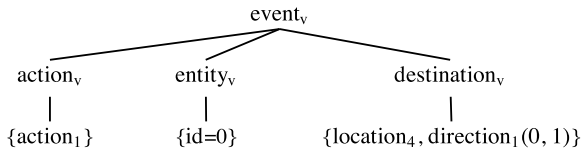


Figure 16: The vision tree extracted from the video in Figure 15.

The procedure to perform the search for the correct RCL tree Ψ 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 16, and show how OLAV obtains a correct RCL tree Ψ from this input video-sentence pair.

6.3.1. Substitute words with visual concepts

For each input sentence S consisting of t words, $S = \langle w_1, \ldots, w_t \rangle$, OLAV substitutes each word with its learnt visual concept using the grounding function Φ. For instance, the sentence $S = \langle$'place', 'the', 'green', 'sphere', 'over', ' the', 'red', 'cube'$\rangle$, is transformed using the grounding function Φ into $S' = \langle action_1, None, colour_2, shape_3, direction_1, None, colour_3, shape_1 \rangle$. The grounding function Φ for this example is shown in Figure 17 (left). Note that if a word has multiple groundings in Φ, then this process is repeated for all combinations of possible groundings.

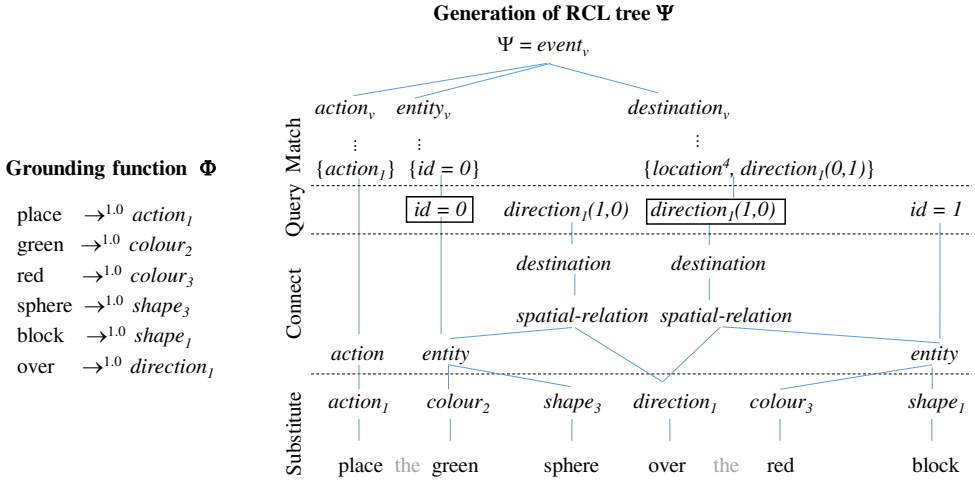


Figure 17: Automatic generation of RCL trees. (left) The grounding function Φ showing the probabilities of assigning words to vision concepts. (right) The four steps (Substitute, Connect, Query, and Match) to generate an RCL tree Ψ from the sentence 'place the green sphere over the red cube'.

6.3.2. Group concepts to generate RCL elements

Once the sentence S is transformed into a list of visual concepts S', OLAV groups the visual concepts in S' to create all possible *entity, action, spatial-relation*, and *destination* RCL elements. The grouping of these elements is performed by connecting the children to generate an RCL element, for example group consecutive *colour*, *shape*, and *location* concepts to form *entity* RCL elements. 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 to English, as is the case in Arabic, or French.

6.3.3. 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 list of predicates for this example is shown in Figure 15. For example, to link the *entity* element $entity(colour_2, shape_3)$, OLAV queries 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 15, OLAV can see that the green sphere, the object with $id = 0$, is the only object that satisfies both constraints. Therefore, it is linked to $id = 0$, and hence has successfully linked part of the input sentence. 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.

6.3.4. Match RCL elements with Ω

OLAV aims to find the correct language structure Ψ by matching the query results to the elements of the vision tree Ω. This is achieved by comparing the values of each RCL element with the vision elements. For example, the vision tree Ω in Figure 16 has an $entity_v$ element with *id*=0. By matching this with the available RCL elements OLAV finds 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 17. By looping through all available options, OLAV matches all elements in the vision tree. Note that any ambiguity represented by a list of ids being returned in Section 6.3.3 is resolved in this matching process. The robot now has the correct sentence structure that reflects what happened in the input video. The resultant sentence structure Ψ from this example is shown in the Figure 18. The names used in the tree (*colour, shape, spatial-relation, etc.*) are used for simplicity and readability. The robot is not assumed to know these words specifically. In the following section, the learning of probabilistic grammar rules using this tree is discussed.

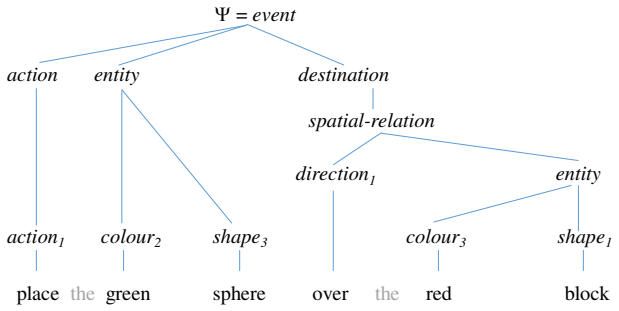


Figure 18: The generated RCL tree Ψ from the example shown in Figure 15.

6.4. Learning Grammar Rules

Grammar induction refers to the process of learning a formal grammar, usually as a collection of re-write rules 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 work, Probabilistic Context Free Grammar (PCFG; also known as Stochastic CFG) is used to model the grammar rules of language. A 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. OLAV learns the PCFG rules by mapping natural language commands to visual features. The main contribution in our grammar induction approach is that OLAV can automatically generate training examples similar to those annotated by human experts to enable the learning of grammar without direct supervision.

6.4.1. Learning a PCFG Π from an RCL tree Ψ

OLAV induces a probabilistic grammar $\Pi = (N, T, R, S, P)$ from an automatically generated language tree (Ψ) such as the one shown in Figure 18. OLAV follows the Inside-Outside algorithm presented by Lari and Young [69] to induce PCFG rules. There are only two kinds of productions in the grammar rules OLAV learns: the non-terminal ones ($\mathcal{B} \rightarrow \mathcal{C}_1, \ldots, \mathcal{C}_m$), and the terminal ones ($\mathcal{B} \rightarrow \mathcal{Z}$), where $\mathcal{B}$ and $\mathcal{C}_i$ are non-terminal symbols, while $\mathcal{Z}$ is a terminal symbol. A probability, denoted $P(\mathcal{C}_1, \ldots, \mathcal{C}_m|\mathcal{B})$ or $P(\mathcal{Z}|\mathcal{B})$, is associated to each production. The computation of these probabilities is shown in Equations 8 and 9, where P is the probability of the grammar rule, ζ 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 10.

$$P(\mathcal{C}_1, \ldots, \mathcal{C}_m|\mathcal{B}) = \frac{\zeta(\mathcal{B} \rightarrow \mathcal{C}_1, \ldots, \mathcal{C}_m)}{\zeta(\mathcal{B} \rightarrow *)} \tag{8}$$

$$P(\mathcal{Z}|\mathcal{B}) = \frac{\zeta(\mathcal{B} \rightarrow \mathcal{Z})}{\zeta(\mathcal{B} \rightarrow *)} \tag{9}$$

$$\sum_{c} P(c_1, \ldots, c_m|\mathcal{B}) + \sum_{z} P(z|\mathcal{B}) = 1 \tag{10}$$

To learn the grammar rules OLAV starts with an empty PCFG rule set. The rules learned from the example sentence 'place the green sphere over the red cube' are shown in Table 3. The rules learned from all examples are accumulated into one PCFG 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 3: The learned grammar rules from the example sentence 'place the green sphere over the red cube' are on the left side, while the probability of each rule is shown to the right. The probability calculations assume that this is the very first sentence considered by the grammar induction component.

6.5. Assumptions and Limitations

Our grammar induction approach has the potential to be expanded to more domains such as robot navigation or cooking recipes, but more RCL elements would have to be manually defined and provided to OLAV; we believe the numbers of new elements would be relatively small, since the elements we already have are quite abstract, but the number of new elements that would be required needs to be determined

by future research[9]. It may even be possible that some of these could be learned rather than given. In any case, 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.

7. Experimental Procedure

OLAV's learning ability is evaluated in four different experiments tackling the: (1) incremental learning of visual concepts from video inputs. (2) incremental language groundings of n-grams to visual concepts. (3) incremental induction of grammar rules, and finally (4) scalability of OLAV's learning framework.

7.1. Datasets

Three different datasets were collected or extended and used to evaluate the performance of the OLAV framework[10]. The datasets involve three robot manipulators performing different table-top tasks such as picking up and moving objects. The three datasets are presented in more detail in the following sections. In all cases, all objects involved remain in the scene throughout.

7.1.1. Extended Train-Robots Dataset

Extended Train-Robots is a simulation dataset (henceforth ETR) 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 [66]. 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. 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. The original dataset contained two shapes only (cube and prism), and eight different colours (red, green, blue, cyan, grey, white, yellow and pink).

In this work, the original dataset[11] has been 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. We modified the scenes to include two more objects (sphere, cylinder) and one more colour (black). This was achieved by changing half the scenes that contain 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. Particular care was taken in modifying the annotated commands to match the scenes in order not to alter the meaning or any mistakes in the descriptions. The commands were manually changed by the first author. The second extension 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 19. The third extension is the translation of the commands from English to Arabic to test OLAV's learning framework on a different language. The translation was performed using the Goslate library[12]. Again, particular care was taken not to alter the commands or correct any mistakes before translation. There was no verification of the translated sentences; this was deliberate since the original English sentences that had been obtained via Amazon Turk were often not grammatical and we did not want to correct one but not the other. Our main purpose in this experiment was not to produce a high quality Arabic grammar, but rather to demonstrate the portability of the framework.

[9] An example of an additional RCL element that would likely be required in the cooking domain is object orientation (since the orientation of objects such as containers is important).

[10] Evaluation on a fourth data set which also includes perception of humans and their activities can be found in [52, 6]

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

[12] goslate 1.5.1, https://pypi.org/project/goslate/ – last accessed December 2018.

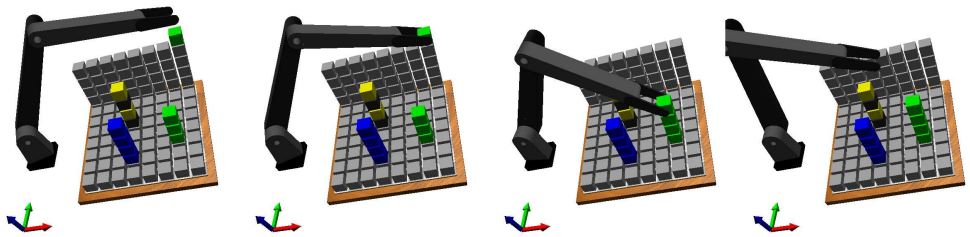


Figure 19: Examples from the ETR dataset along with their annotated commands; the Arabic sentences are automatically translated from the English ones.

7.1.2. Leeds Robotic Commands Dataset

The Leeds Robotic Commands dataset (henceforth LRC)[13] was first presented in the earlier conference paper this article extends, Alomari *et al.* [4]. The dataset contains real-world RGB-D scenes of a robot manipulating different objects together with natural language descriptions of these actions. We used a Baxter robot (shown in Figure 20) from Rethink Robotics fitted with a Microsoft Kinect2 sensor on its chest such that it can observe and model its environment in RGB-D. To perform each task, a demonstrator was asked to drive the robot using a joystick[14]. Only one object is manipulated in each video. The three commands used in this dataset to guide the demonstrator are 'pick up', 'put down' and 'move'. The dataset includes 204 video clips containing a total of 51 different objects including basic block shapes, fruits, cutlery, and office supplies, with a mean of five objects present in each scene. The detected objects are shown in the bottom row of Figure 20 where each object is assigned a unique *id* and tracked throughout the video.

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 the robot, 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, a mean of five per video as in the ETR dataset.

7.1.3. Extended Object Ordering Dataset

The original Object Ordering dataset was presented by Sinapov *et al.* [48], and was designed to teach a robot to arrange objects in an ascending order based on their properties. For example, to arrange objects from shortest to tallest, smallest to largest, etc. To learn about object properties, the robot performs seven different actions on each object in the scene. The actions are *grasp, lift, lower, drop, press, push* and *hold*. The set of objects that the robot explores and learns about consists of 32 common household items including cups, bottles, cans, and other containers, with variation in weight, height, and width. Also, each video clip features only a single object in it, which means the robot can not 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, henceforth named the EOO dataset. The commands were provided by annotators who viewed the video clips one at a time. Given the simplicity of the EOO environment (only one object in each scene), it was decided that a single annotation per video was sufficient. The dataset contains a total of 1120 video clips, each of which was annotated with a linguistic command. An example of video clip and its corresponding command is shown in Figure 21. Next, we describe the four experiments.

7.2. Experiment 1: Learning Visual Concepts

We present here the empirical results which comprise the evaluation of the visual concept learning framework. Since the learning is performed in a loosely-supervised setting, and OLAV does not know the label

[13]The LRC dataset is available at http://doi.org/10.5518/110
[14]The Python and ROS implementation are available at https://github.com/OMARI1988/baxter_pykdl

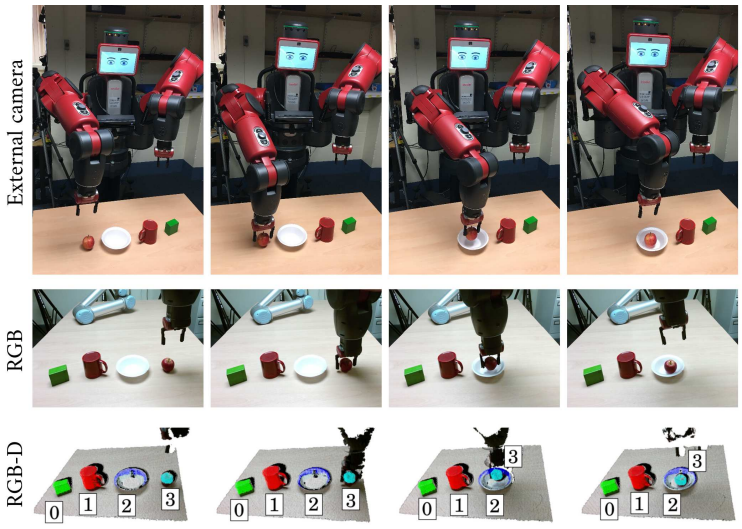


Figure 20: Example from the LRC dataset for the command 'move the red apple into the white bowl'. (top) An external camera is placed opposite the robot to record the scene. Note that this camera is not used in object detection nor tracking. (middle) The RGB feed from the Kinect2 sensor showing the point-of-view of the robot. (bottom) The RGB-D feed from Kinect2, along with the detected objects' *id*s and tracks.

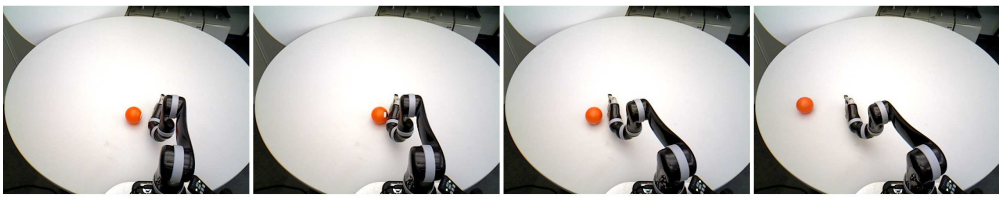


Figure 21: An example from the EOO dataset along with its annotated natural language command.

of each concept, we use two clustering metrics to evaluate the performance: normalised *Mutual Information* [70], and *V-measure* [71].

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 has access to the ground truth labels during training; nevertheless we show that the SVM only marginally outperforms OLAV in the three datasets. Figure 22 presents the results of OLAV's incremental, loosely-supervised visual concept extraction on all three datasets. Examples of learned visual concepts are presented in Figure 23. A detailed analysis of the obtained results is presented below.

The number of learned concepts was selected unsupervised using a BIC and graph matching approaches. For example, in the LRC dataset, the robot thinks there are 25 unique shape concepts in this dataset, when in fact there are only 13 classes. 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 to identify the individual objects in the scene does not produce perfect object segments, which leads to having objects with incorrect point cloud segments (with extra or missing points/parts). Second, using a particle filter to track objects 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 laboratory

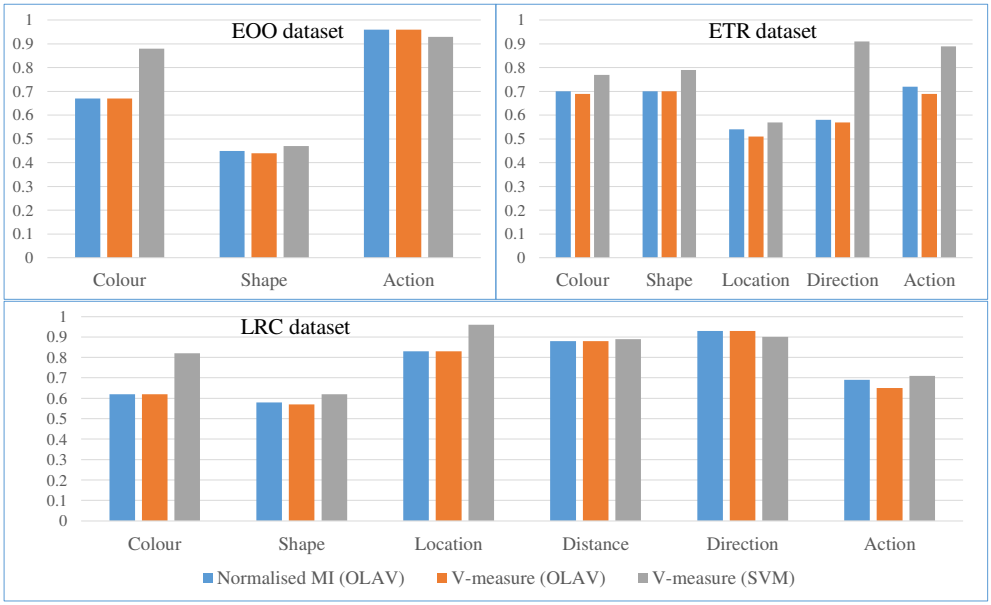


Figure 22: Experimental results of visual concept extraction for all three datasets, showing two clustering metrics (Normalised Mutual Information, and V-measure) for colour, shape, location, direction, distance, action. Note that not all feature spaces are applicable to every dataset. The V-measure for a supervised method (SVM) is also shown for comparison.

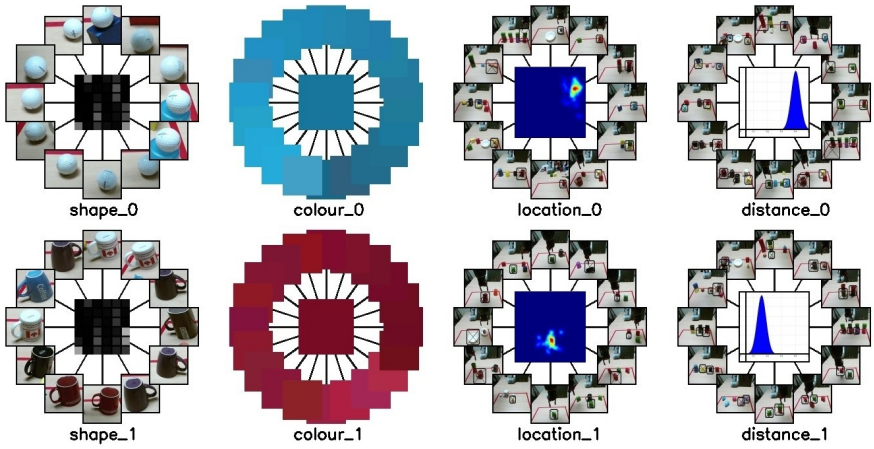


Figure 23: Examples of visual concepts learned from the LRC dataset. The centre image shows the geometric median for the shape and colour examples, and the distribution of the cluster for the direction and distance examples; the surrounding images show samples from each cluster.

which led 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, OLAV still managed to learn and cluster the visual concepts with comparable accuracy to the supervised SVM system, albeit lower in most cases. It produced a better result in two cases (direction in the LRC dataset and action in the EOO dataset).

The results obtained from all three datasets show that OLAV is capable of learning visual concepts from robot observation in a loosely-supervised online setting. These learned visual concepts are used in the

following experiment to learn language groundings.

7.3. Experiment 2: Learning Language Groundings

The grounding results for all three datasets are obtained using n-grams of length less than or equal to three. For ground truth, we manually annotated all correct word-vision groundings for each of the learned visual concepts in the three datasets, e.g. the word 'red' should be grounded to the learned Gaussian component of the colour red, The grounding was learnt in an online fashion, i.e. each video-sentence pair was processed and then discarded, retaining only the updated model. No attempt was made to optimise the ordering of the inputs (which might happen if a human trainer was supervising the learning explicitly). As a metric, we compute the F1-score [72] of the grounding results in each feature space separately.

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 [73]. 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 HMM requires 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*) for learning. A four fold cross validation is performed to compute the F1-scores for the HMM system on all three datasets and the mean of these is presented. Figure 24 presents the final (i.e. after all video-sentence pairs have been processed) F1-scores computed using our online learning framework and the supervised HMM system for each of the three datasets[15]. The results show that OLAV was able to successfully learn part of the correct language groundings in each dataset and how OLAV compares with the supervised HMM system, keeping in mind that the HMM has access to the ground truth labels to learn from, whilst OLAV learns from unlabelled sentences.

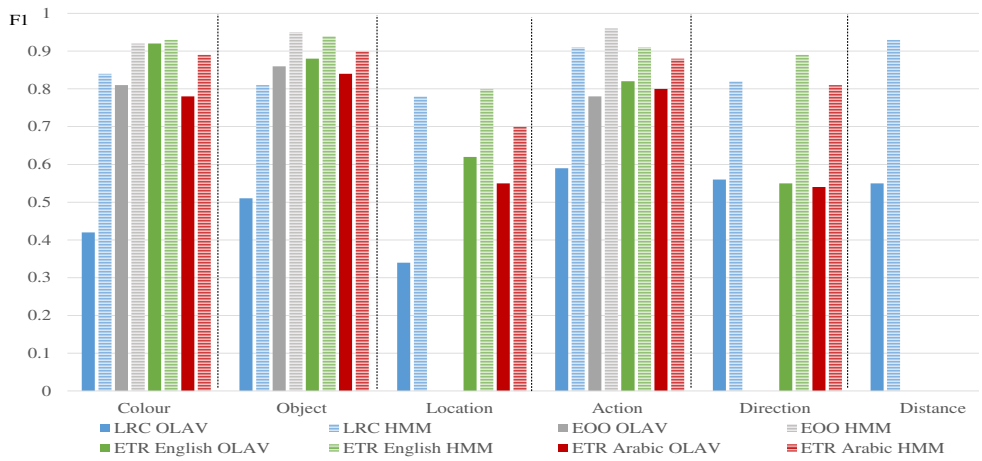


Figure 24: Natural language grounding results. The solid bars represent OLAV's F1 performance, and the adjacent hatched bar immediately to the right represents the F1 performance of an HMM based supervised upper bound. It can be seen that in many cases the performance of OLAV approaches the performance of the HMM system. Not all feature spaces are present in each dataset, so there are varying numbers of bars (only LRC has all feature spaces). (Best viewed in colour.)

Figure 25 shows the language grounding incremental results obtained using OLAV from each of the three datasets. The graphs show an improving trend in the F1-score of the word groundings in each feature space as more data is observed and 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.

[15]Note that the final two columns relate to results obtained in the Arabic version of the ETR dataset which will be discussed in Section 7.3.1

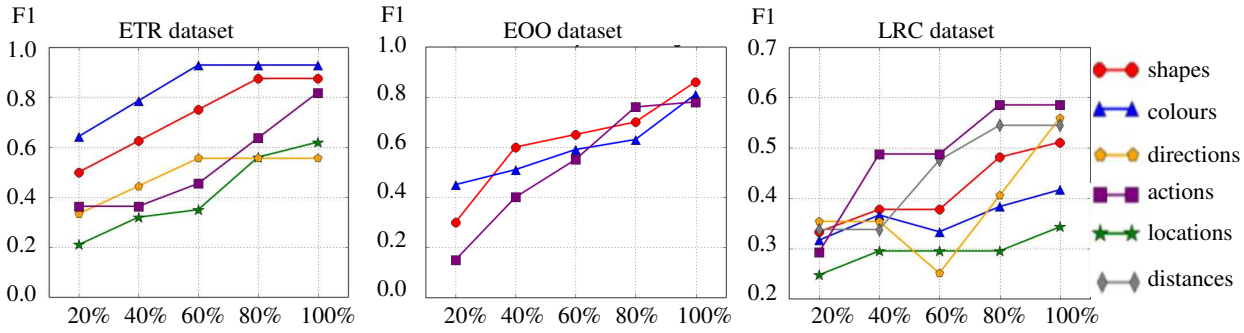


Figure 25: Incremental language grounding. F1-scores for incremental English language grounding for each dataset. Note that a different y-axes scale was selected for the LRC dataset to better show the results.

7.3.1. Grounding in other languages

In this section we evaluate our language grounding framework in learning from other languages. We use the translated commands in the ETR dataset for this evaluation. The learning framework is applied on the Arabic language in the exact same way as the English language. Figure 24 presents the results of language grounding in both Arabic and English for the ETR Dataset. As an upper bound, we again present the results obtained using a supervised Hidden Markov Model (HMM) for POS tagging.

The results in Figure 24 show that OLAV 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 Arabic has two genders (masculine and feminine) which have different lexical forms (e.g. masculine grey → ramady, feminine grey → ramadia) and since there is no notion of stemming built into OLAV, each is treated as a separate word to learn, effectively reducing the number of training examples (e.g. masculine grey objects vs feminine grey objects). With this in mind, OLAV still managed to ground words in Arabic to their corresponding visual concepts. Examples of learned groundings from both Arabic and English are shown in Figure 26. The arrows are used to indicate the direct translation between the two words. This means that OLAV can be used to learn translation between languages based on their groundings to the visual domain, but we leave this idea open for future work to investigate and validate.

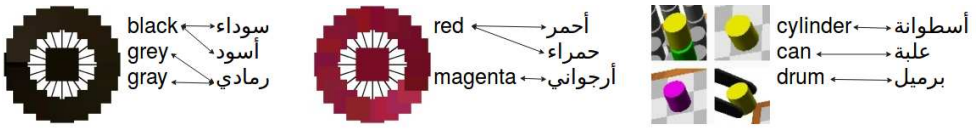


Figure 26: Examples of learned language groundings from both Arabic and English in the ETR dataset. The training was performed on each language separately. The arrows between words are used to indicate the direct translation between the two words and were manually added to the image. OLAV does not know that these words are translations in different languages, though this would be a natural extension to investigate as noted in the main text.

7.3.2. Sensitivity analysis for grounding the parameter (ϵ)

As described in Section 5.4, ϵ is used as a threshold to keep groundings relatively sparse. The selection of a value for ϵ was based on a sensitivity analysis experiment performed over four datasets[16]. The results of this experiment (Figure 27) show that the grounding performance peaks at $\epsilon = 0.05$ for most of the feature spaces in the datasets, and therefore this value was selected for all the grounding experiments.

[16]We performed the analysis on the three datasets in this paper (ETR, EOO and LRC) as well as a fourth one consisting of data recorded in a kitchen scene[52] which was included in the thesis [6] from which this paper originated. Note that the sensitivity analysis was not performed on a separate validation set, which might limit generalisability.

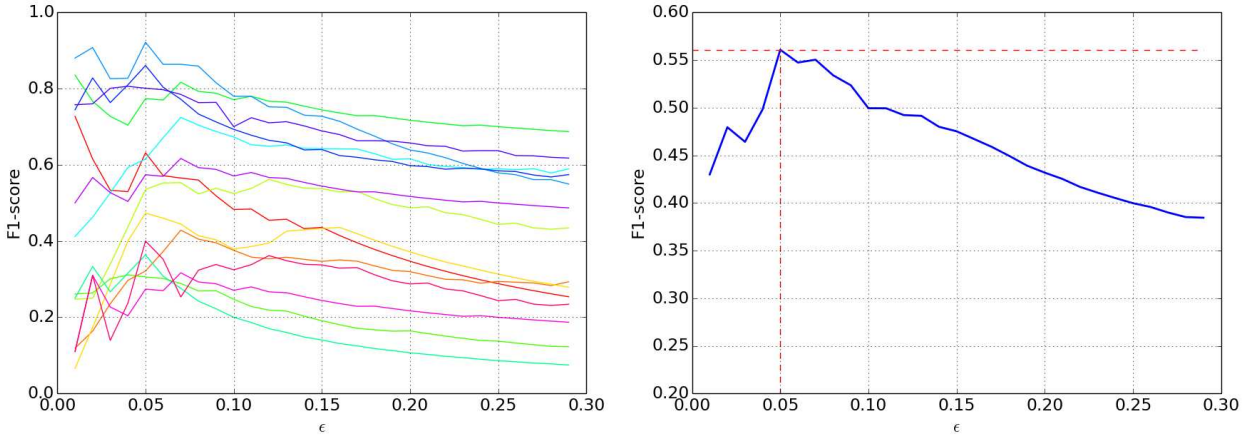


Figure 27: Sensitivity analysis for the language grounding parameter epsilon (ϵ). (left) the graph shows the final F1-score values in each feature space from the datasets on the y-axis, and the different ϵ values used to compute these F1-scores on the x-axis. (right) the (mean) F1-score results obtained from all feature spaces.

7.4. Experiment 3: Learning Grammar Rules

In this subsection we evaluate our grammar induction framework based on its ability to learn grammar rules capable of parsing never-seen-before linguistic commands.

To evaluate our grammar induction framework and the learned grammar (Π), we test it on the three datasets. Each of the three datasets is randomly divided into four folds, and four fold cross-validation is applied. 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 folds. 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 [74]. This supervised system has access to the human annotated RCL trees to learn the grammar rules from, while OLAV automatically generates them. The same four fold cross validation procedure is applied on this supervised system.

We also tested OLAV against an unsupervised grammar induction approach presented by Ponvert *et al.* [67] which learns a language model via chunking the raw text into smaller parts that show a repeated pattern throughout the dataset. This represents an unsupervised baseline. Both OLAV and Ponvert's learn from unlabelled sentences. However, OLAV learns from language and vision inputs, while Ponvert's system learns from language alone. We evaluate the baseline (Ponvert's unsupervised system) based on its ability to chunk the text into correct sub-trees only, as it does not generate labels.

Figure 28 presents the grammar induction results for the three systems (i) Abney's supervised system, (ii) OLAV, and (iii) the baseline across each of the three datasets[17]. The results clearly show that OLAV outperforms the baseline. The number of grammar rules generated differs between techniques as shown at the top ofb Figure 28. The supervised rules are higher in number because a few sentences contain classes which OLAV can not learn (i.e. OLAV fails to generate a grammar tree from the input sentence). For example, in the ETR dataset there exists an indicator class for superlatives, e.g. (indicator $\rightarrow^{w}$ tallest), but as already

[17]Note that the final column relates to results obtained in the Arabic version of the ETR dataset which will be discussed in Section 7.4.1.

noted OLAV cannot handle superlatives yet. However, the results do not vary as much because there are not many sentences containing such classes in our three datasets.

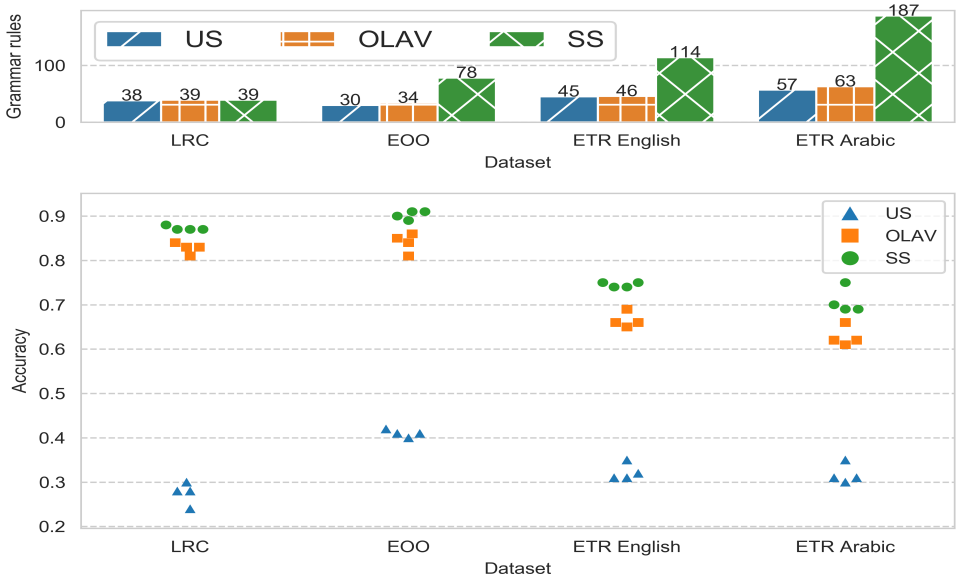


Figure 28: Grammar induction results. US stands for baseline unsupervised system ([67]), while SS stands for the supervised system ([74]). (top) The mean number of grammar rules or productions generated in all four folds. OLAV generates approximately the same number of rules as US, and many fewer than SS (except for the LRC dataset), showing better generalisation. (bottom) The plotted values are the fraction of correctly parsed sub-trees in each of four test folds. (Best viewed in colour.)

An example from one of the test commands in the ETR dataset is presented in Figure 29. The example is for the command 'place the yellow ball on top of the blue cylinder'. The figure shows the parse tree using the learned grammar set (II) from our approach (top), and the parsed tree using the baseline of Ponvert's unsupervised system that learns from language alone (bottom). The learned grammar rules from OLAV used to parse this natural language command are presented in Table 4. For example, the grammar rule (colour $\rightarrow^{0.16}$ *yellow*) is used to tag the word *yellow* as a colour terminal symbol, similarly the rule (shape $\rightarrow^{0.13}$ *ball*) is used to tag the word *ball* as a shape terminal symbol, 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 (Cocke-Younger-Kasami algorithm [75]) 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).

We also plotted the rate of grammar acquisition as successively larger percentages of the training set were used – see figure 30. For this experiment, 25% of the data was reserved as a test set, and then successively larger percentages of the data was used to learn a grammar. We ran this experiment three times, randomly selecting the initial sentences, and the incremental additions using 3 different seeds. Since the learning rate rises rapidly, we added 5 sentences each time until 250 sentences were reached, and then added 15 sentences each time. It can be seen that a grammar can be learnt quite quickly, with less than 20% of the sentences, i.e. less than 1000 sentences.

7.4.1. Grammar induction in other languages

In this subsection we evaluate our grammar induction framework on the Arabic version of the ETR. The learning of grammar rules is applied to the Arabic language data in the exact same way as the English

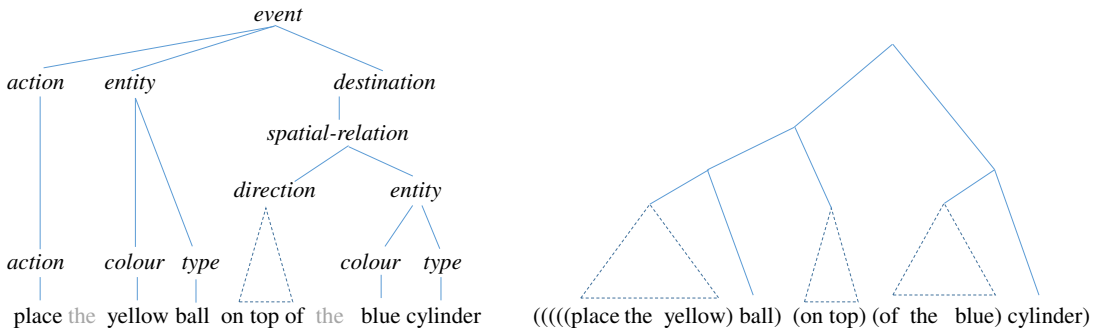


Figure 29: The grammar trees generated for the new command 'place the yellow ball on top of the blue cylinder' using OLAV (left) and baseline of Ponvert's unsupervised system (right).

Terminal Leaves	*Non-Terminals*
colour $\rightarrow^{0.16}$ 'yellow'	event $\rightarrow^{0.33}$ action, entity, location
colour $\rightarrow^{0.22}$ 'blue'	entity $\rightarrow^{0.16}$ colour, shape
shape $\rightarrow^{0.13}$ 'ball'	location $\rightarrow^{0.81}$ 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 4: The learned grammar rules used to parse the command 'place the yellow ball on top of the blue cylinder' shown in Figure 29.

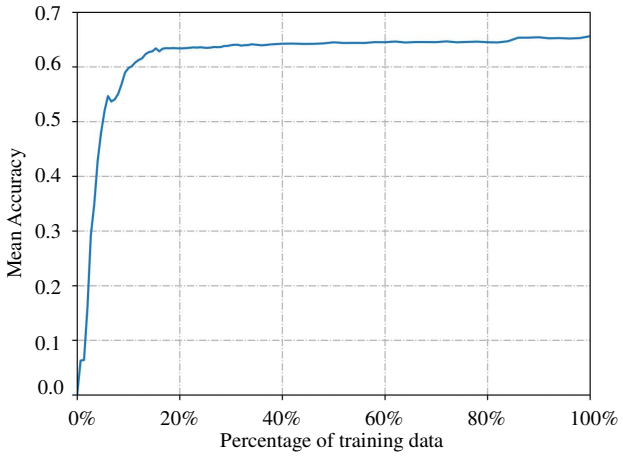


Figure 30: Incremental grammar induction results. A plot showing the rate of grammar acquisition with increasing number of sentences in the training set. Results are averaged over 3 random folds, with the same 250 sentences used for testing in all folds and all training set sizes.

language data.

The final column of Figure 28 presents the results of grammar induction in Arabic for the ETR dataset. We again compare the results from OLAV against the same unsupervised lower bound baseline, and the

supervised upper bound. The results in Figure 28 show that our approach outperforms the unsupervised baseline 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 OLAV is capable of learning grammar rules regardless of the POS tags ordering in a sentence – for example, in Arabic adjectives come after the noun.

7.5. Experiment 4: Scalability and Memory Requirements

In this subsection, we present empirical results to evaluate the scalability of our language and vision learning framework. 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 work for teaching robots about language and vision.

We evaluate the scalability of the three main components in OLAV, (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 calculations were performed on a desktop PC, with an Intel Core i7-4790 processor with 8 cores, 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. Figure 31 shows the incremental memory requirement of the three components in OLAV along with the raw size of the input data in the LRC datasets. The graphs in Figure 31 show how efficient OLAV 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 LRC 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 OLAV has learned most of the visual and linguistic concepts there are to learn in this dataset. Most of the vision concepts, word groundings, and grammar rules have been observed and allocated a location in memory. We hypothesise that extended observation of the environment will scale well in OLAV as the size of the learned models will not increase linearly with the size of observed data, but rather will flatten as OLAV incrementally learns everything it is capable of discovering from a dataset.

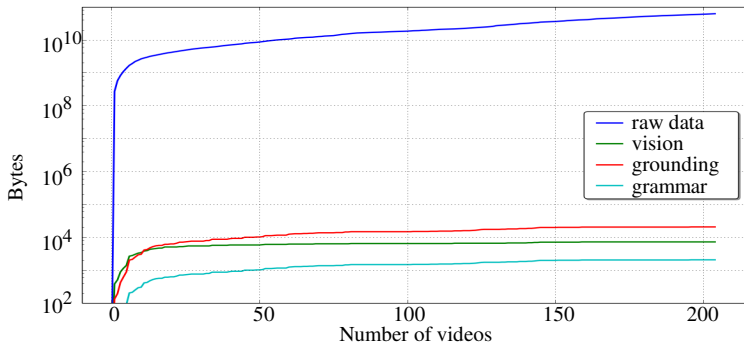


Figure 31: The incremental memory requirement of OLAV on the LRC dataset.

Finally, we briefly discuss the computational requirements imposed by the use of ILP, which is known to be computationally hard[18]. However this has not proved a problem in our experiments, (e.g. the

[18] 0-1 integer linear programming is one of Karp's 21 NP-complete problems [76].

entire dataset for the largest dataset (ETR), was processed in under an hour (∼5000 video-sentence pairs). The habituation step, which removes variables, also helps from a theoretical viewpoint. If this proved a problem, then using a solver which can exploit the previous solution when presented with a slightly different matrix/objective function is likely to be beneficial (e.g [77]). Another approach would be to use heuristic methods[78], as a good solution rather than an optimal solution is likely to be sufficient.

8. Conclusion and Future work

We have presented a novel, incremental and loosely-supervised framework that enables robots to bootstrap their knowledge in language and vision domains by incrementally learning three kinds of knowledge:

- Visual representations of the world in a number of predefined feature spaces.
- Language grounding that maps phrases in language to their corresponding learned visual concepts.
- Probabilistic grammar rules of natural language.

Previous systems were designed to use one or two of these three components (visual representation, language grounding and language grammar) to learn the remaining one(s). To the best of our knowledge, this is the first system capable of learning all three components, which thus reduces the amount of needed initial knowledge significantly. Macroscopically, viewing the processing of each video-sentence pair as a single atomic operation, OLAV can be viewed as jointly or concurrently learning the three components. Also, we show that these components can be learned from real-world noisy data collected using robots equipped with different sensing modalities, and in different languages (Arabic and English). We also offer a number of individual contributions in the fields of visual learning, language grounding, and grammar induction.

8.1. Visual learning

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 OLAV assumes no prior knowledge of the structure of the sensor feature spaces, it employs probabilistic modelling techniques to each feature space independently to elicit meaningful classes that are supported by the observed data.

One of the key novel contributions we offer in this field is the use of incremental Gaussian mixture models and a BIC to learn the simple visual concepts in a loosely-supervised manner. Notably, OLAV is also able to learn not just unary predicates (colours, shapes, locations), but also binary relations (relative directions and distances). The extended spatio-temporal graphs (STDAG) representation is also a key novel contribution of our work acting as an intermediate representation between the continuous perceptual space, and the purely symbolic linguistic structures, enabling the learning of complex visual concepts.

8.2. Language grounding

Language grounding is the second step in our learning framework, and is performed after updating the visual concepts in each video clip. OLAV searches 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.

8.3. Grammar induction

Grammar induction is the third and final step in our learning framework. The main novel contribution we offer in the field of grammar induction is that OLAV *automatically* generates training examples similar to those annotated by a human expert. This is achieved by utilizing the learned groundings and the extracted vision trees to successfully replace the human annotator. OLAV searches the space of all possible language trees from a sentence for one that matches the extracted vision tree. An important advantage of the use of the RCL trees is that they provide a semantics which would enable a robot to execute a newly parsed sentence.

8.4. 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 streams of audio-video data or YouTube videos. First, it requires the videos and sentences to be (roughly) 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).

8.4.1. Learning from non-segmented videos and text

Providing our robots with the ability to learn from long, non-segmented videos and text would likely significantly improve the learning. This would 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, OLAV could be upgraded using an idea similar to that presented by Alayrac *et al.* [79]. In their work, they presented a system capable of automatically learning the main steps to complete a given task, such as changing a car tyre, 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'.

8.4.2. Generating new visual and relational features

Visual features are the representation or encoding used to move from pixel level inputs into a space where visual concepts can be learned. These feature spaces are manually defined in this work, such as the HSL colour feature space. The manual identification 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 would enable our robot to learn more visual concepts without the need to manually define each feature space. It is possible that representations such as Eigenobject learned vectors [80] which learn a generative model for object classes could be useful in this context. But this still leaves open the question of new relational feature spaces; for example the present feature spaces cannot explicitly encode topological information, such as *x is inside y.* One possible way of addressing this problem is to create a set of primitive features, which can be used to generate new feature spaces as presented by Bennett *et al.* [81]. 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.

8.5. Generalising constraints

The objective function (formula 2) used in the ILP maximisation process is subject to two constraints. The first of these is used to keep groundings relatively sparse by setting a threshold. This has been found to be effective, and the sensitivity of the threshold was analysed in section 7.3.2. However it might be possible to have a more elegant solution involving an exponential penalty. However the present limited size of the datasets mitigated against such a solution, but could be investigated in larger scale experimentation. Similar comments apply to the filters (3-6) used to filter incorrect n-grams – an exponential filter that favours lower n-gram assignments might provide a more elegant solution, but again was not investigated since we felt a rather larger dataset would be needed for this to be effective. In practice the constraints as presented in the paper have been found to work well.

9. Acknowledgements

We thank colleagues in the School of Computing Robotics lab, in particular Majd Hawasly, Paul Duckworth, and the STRANDS project consortium (http://strands-project.eu) for their valuable comments. We also acknowledge the financial support provided by EU FP7 project 600623 (STRANDS). We also acknowledge the financial support under grant agreement 825619 (AI4EU). The third and fourth authors were partially supported by Fellowships from the Alan Turing Institute. We also thank the anonymous referees and the Associate Editor whose comments on the earlier versions have helped improve the paper.

References

[1] M. Hasan, M. Warburton, W. C. Agboh, M. R. Dogar, M. Leonetti, H. Wang, F. Mushtaq, M. Mon-Williams, A. G. Cohn, Human-like planning for reaching in cluttered environments, in: 2020 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2020, pp. 7784–7790.

[2] M. Alomari, E. Chinellato, Y. Gatsoulis, D. C. Hogg, 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, 2016, pp. 505–508.

[3] M. Alomari, P. Duckworth, D. C. Hogg, 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), 2017.

[4] M. Alomari, P. Duckworth, D. C. Hogg, 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, 2017, pp. 4349–4356.

[5] M. Alomari, P. Duckworth, Y. Gatsoulis, D. C. Hogg, 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., 2016.

[6] M. Al-omari, Joint perceptual learning and natural language acquisition for autonomous robots, Ph.D. thesis, School of Computing, University of Leeds, UK (2017).
URL `https://etheses.whiterose.ac.uk/18860/`

[7] T. Winograd, Understanding natural language, Cognitive Psychology 3 (1) (1972) 1–191.

[8] Y. Bisk, A. Holtzman, J. Thomason, J. Andreas, Y. Bengio, J. Chai, M. Lapata, A. Lazaridou, J. May, A. Nisnevich, N. Pinto, J. Turian, Experience grounds language, in: Empirical Methods in Natural Language Processing (EMNLP), 2020.

[9] M. Spranger, K. Beuls, Referential uncertainty and word learning in high-dimensional, continuous meaning spaces, in: 2016 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics, ICDL-EpiRob 2016, Cergy-Pontoise, France, September 19-22, 2016, IEEE, 2016, pp. 95–100.

[10] M. Quine, Word and object, MIT Press, 1960.

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

[12] H. Yu, H. Zhang, W. Xu, Interactive grounded language acquisition and generalization in a 2D world, in: International Conference on Learning Representations, 2018.

[13] C. J. Needham, P. E. Santos, D. R. Magee, V. Devin, D. C. Hogg, A. G. Cohn, Protocols from perceptual observations, Artificial Intelligence 167 (1) (2005) 103–136.

[14] S. Pezzelle, I.-T. Sorodoc, R. Bernardi, Comparatives, quantifiers, proportions: a multi-task model for the learning of quantities from vision, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 419–430.

[15] T. Rahgooy, U. Manzoor, P. Kordjamshidi, Visually guided spatial relation extraction from text, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 788–794.

[16] H. Yu, N. Siddharth, A. Barbu, J. M. Siskind, A compositional framework for grounding language inference, generation, and acquisition in video, Journal of Artificial Intelligence Research 52 (2015) 601–713.

[17] E. H. Huang, R. Socher, C. D. Manning, A. Y. Ng, Improving word representations via global context and multiple word prototypes, in: ACL, ACL, 2012, pp. 873–882.

[18] D. Roy, B. Schiele, A. Pentland, Learning audio-visual associations using mutual information, in: Proceedings Integration of Speech and Image Understanding, IEEE, 1999, pp. 147–163.

[19] J. Sinapov, C. Schenck, A. Stoytchev, Learning relational object categories using behavioral exploration and multimodal perception, in: 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 5691–5698. `doi:10.1109/ICRA.2014.6907696`.

[20] J. Thomason, J. Sinapov, M. Svetlik, P. Stone, R. J. Mooney, Learning multi-modal grounded linguistic semantics by playing "I spy", in: International Joint Conference on Artificial Intelligence (IJCAI), 2016, pp. 3477–3483.

[21] J. Thomason, J. Sinapov, R. Mooney, Guiding interaction behaviors for multi-modal grounded language learning, in: Proceedings of the First Workshop on Language Grounding for Robotics, Association for Computational Linguistics, Vancouver, Canada, 2017, pp. 20–24.

[22] O. A. Can, P. Z. D. Martires, A. Persson, J. Gaal, A. Loutfi, L. De Raedt, D. Yuret, A. Saffiotti, Learning from implicit information in natural language instructions for robotic manipulations, arXiv abs/1904.13324.

[23] M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mosenlechner, D. Pangercic, T. Ruhr, M. Tenorth, Robotic roommates making pancakes, in: Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on, IEEE, 2011, pp. 529–536.

[24] S. Tellex, N. Gopalan, H. Kress-Gazit, C. Matuszek, Robots that use language, Annual Review of Control, Robotics, and Autonomous Systems 3 (1) (2020) 25–55. doi:https://doi.org/10.1146/annurev-control-101119-071628.

[25] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, D. Fox, ALFRED: A benchmark for interpreting grounded instructions for everyday tasks, in: Computer Vision and Pattern Recognition (CVPR), 2020, pp. pp. 10740–10749.

[26] K. Dukes, Semeval-2014 task 6: Supervised semantic parsing of robotic spatial commands, SemEval 2014 (2014) 45.

[27] S. I. Wang, P. Liang, C. D. Manning, Learning language games through interaction, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2016, pp. 2368–2378.

[28] C. Ross, A. Barbu, Y. Berzak, B. Myanganbayar, B. Katz, Grounding language acquisition by training semantic parsers using captioned videos, in: Proc. Conference on Empirical Methods on Natural Language Processing, (EMNLP), 2018, pp. 2647–2656.

[29] C. Matuszek, N. Fitzgerald, L. Zettlemoyer, L. Bo, D. Fox, A joint model of language and perception for grounded attribute learning, in: Proceedings of the 29th International Conference on Machine Learning (ICML 2012), 2012, p. 1435–1442.

[30] C. Kerry, Esta es una naranja atractiva: Adventures in adapting an English language grounding system to non-English data, Master's thesis, University of Maryland, Baltimore County (2019).
URL https://iral.cs.umbc.edu/Theses/caroline-kery-ms.pdf

[31] D. L. Chen, R. J. Mooney, Learning to interpret natural language navigation instructions from observations., in: Proceedings of Association for the Advancement of Artificial Intelligence (AAAI), Vol. 2, AAAI Press, 2011, pp. 1–2.

[32] J. Nevens, P. Van Eecke, K. Beuls, From continuous observations to symbolic concepts: A discrimination-based strategy for grounded concept learning, Frontiers in Robotics and AI 7 (2020) 84. doi:10.3389/frobt.2020.00084.
URL https://www.frontiersin.org/article/10.3389/frobt.2020.00084

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

[34] A. S. Huang, S. Tellex, A. Bachrach, T. Kollar, D. Roy, N. Roy, Natural language command of an autonomous micro-air vehicle, in: Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on, IEEE, 2010, pp. 2663–2669.

[35] S. Tellex, T. Kollar, S. Dickerson, M. R. Walter, A. G. Banerjee, S. Teller, N. Roy, Approaching the symbol grounding problem with probabilistic graphical models, AI magazine 32 (4) (2011) 64–76.

[36] C. Matuszek, E. Herbst, L. Zettlemoyer, D. Fox, Learning to parse natural language commands to a robot control system, in: Experimental Robotics, Springer, 2013, pp. 403–415.

[37] D. P. Barrett, S. A. Bronikowski, H. Yu, J. M. Siskind, Driving under the influence (of language), IEEE Transactions on Neural Networks and Learning Systems 29 (7) (2018) 2668–2683.

[38] S. Patki, E. Fahnestock, T. M. Howard, M. R. Walter, Language-guided semantic mapping and mobile manipulation in partially observable environments, in: Proceedings of the Conference on Robot Learning (CoRL), Osaka, Japan, 2019, pp. 1201–1210.

[39] O. Roesler, A. Nowé, Action learning and grounding in simulated human-robot interactions, The Knowledge Engineering Review 34 (2019) e13. doi:10.1017/S0269888919000079.

[40] J. Thomason, A. Padmakumar, J. Sinapov, J. Hart, P. Stone, R. J. Mooney, Opportunistic active learning for grounding natural language descriptions, in: S. Levine, V. Vanhoucke, K. Goldberg (Eds.), Proceedings of the 1st Annual Conference on Robot Learning, Vol. 78 of Proceedings of Machine Learning Research, PMLR, 2017, pp. 67–76.

[41] L. Steels, F. Kaplan, Aibo's First Words: The Social Learning of Language and Meaning, Evolution of Communication 4 (1) (2000) 3–32.

[42] 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, IEEE, 2013, pp. 1640–1647.

[43] L. She, S. Yang, Y. Cheng, Y. Jia, J. Y. Chai, 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, Vol. 89, 2014, pp. 89–97.

[44] L. She, J. Chai, Interactive learning of grounded verb semantics towards human-robot communication, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Vancouver, Canada, 2017, pp. 1634–1644. doi:10.18653/v1/P17-1150.
URL https://www.aclweb.org/anthology/P17-1150

[45] M. Spranger, L. Steels, Co-acquisition of syntax and semantics - an investigation in spatial language, in: Q. Yang, M. Wooldridge (Eds.), Proceedings of International Joint Conference on Artificial Intelligence(IJCAI), AAAI Press, Palo Alto, US, 2015, pp. 1909–1915.

[46] J. Thomason, A. Padmakumar, J. Sinapov, N. Walker, Y. Jiang, H. Yedidsion, J. Hart, P. Stone, R. Mooney, Jointly improving parsing and perception for natural language commands through human-robot dialog, Journal of Artificial

Intelligence Research 67 (2020) 327–374.
[47] D. Roy, Learning Visually Grounded Words and Syntax for a Scene Description Task, Computer Speech & Language 16 (3) (2002) 353–385.
[48] J. Sinapov, P. Khante, M. Svetlik, 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, pp. 3462–3468.
[49] M. Muja, M. Ciocarlie, Tabletop_objects(last accessed 16-12-2016), https://wiki.ros.org/tabletop_objects (2013).
[50] U. Klank, D. Pangercic, R. B. Rusu, M. Beetz, Real-time CAD Model Matching for Mobile Manipulation and Grasping, in: 9th IEEE-RAS International Conference on Humanoid Robots, Paris, France, 2009, pp. 290–296.
[51] R. B. Rusu, Semantic 3D object maps for everyday manipulation in human living environments, Ph.D. thesis, Computer Science department, Technische Universitaet Muenchen, Germany (2009).
[52] M. Alomari, P. Duckworth, N. Bore, M. Hawasly, D. C. Hogg, A. G. Cohn, Grounding of human environments and activities for autonomous robots, in: International Joint Conference on Artificial Intelligence (IJCAI), 2017, pp. 1395–1402.
[53] P. C. Mahalanobis, On the generalized distance in statistics, Proceedings of the National Institute of Sciences (Calcutta) 2 (1936) 49–55.
[54] N. Christofides, Graph Theory An Algorithmic Approach, New York: Academic Press Inc., 1975.
[55] M. Sridhar, A. G. Cohn, D. C. Hogg, Discovering an event taxonomy from video using qualitative spatio-temporal graphs, in: ECAI 2010-19th European Conference on Artificial Intelligence, Proceedings, Vol. 215, IOS Press, 2010, pp. 1103–1104.
[56] M. Sridhar, A. G. Cohn, D. C. Hogg, Unsupervised Learning of Event Classes from Video, in: Proceedings of Association for the Advancement of Artificial Intelligence (AAAI), AAAI Press, 2010, pp. 1631–1638.
[57] Y. Gatsoulis, M. Alomari, C. Burbridge, C. Dondrup, P. Duckworth, P. Lightbody, M. Hanheide, N. Hawes, A. G. Cohn, QSRlib: a software library for online acquisition of Qualitative Spatial Relations from Video, in: Workshop on Qualitative Reasoning, at IJCAI, 2016.
[58] P. Duckworth, Y. Gatsoulis, F. Jovan, D. C. Hogg, 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, 2016, pp. 1043–1051.
[59] J. F. Allen, Maintaining knowledge about temporal intervals, Communications of the ACM 26 (11) (1983) 832–843.
[60] A. G. Cohn, N. M. Gotts, The 'egg-yolk' representation of regions with indeterminate boundaries, in: P. Burrough, A. M. Frank (Eds.), Proceedings, GISDATA Specialist Meeting on Geographical Objects with Undetermined Boundaries, Francis Taylor, 1996, pp. 171–187.
[61] K. S. R. Dubba, A. G. Cohn, D. C. Hogg, M. Bhatt, F. Dylla, Learning relational event models from video, Journal of AI Research.
[62] M. Song, H. Wang, Highly Efficient Incremental Estimation of Gaussian Mixture Models for Online Data Stream Clustering, in: Defense and Security, International Society for Optics and Photonics, 2005, pp. 174–183.
[63] B. S. Everitt, A. Skrondal, The Cambridge dictionary of statistics, University of Cambridge Press: Cambridge, 2002.
[64] E. Howorka, A characterization of distance-hereditary graphs, The Quarterly Journal of Mathematics 28 (4) (1977) 417–420.
[65] K. Sparck Jones, A statistical interpretation of term specificity and its application in retrieval, Journal of documentation 28 (1) (1972) 11–21.
[66] 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.
[67] E. Ponvert, J. Baldridge, 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, Association for Computational Linguistics, Portland, Oregon, USA, 2011, pp. 1077–1086.
[68] P. F. Dominey, J. D. Boucher, Developmental Stages of Perception and Language Acquisition in a Perceptually Grounded Robot, Cognitive Systems Research 6 (3) (2005) 243–259.
[69] K. Lari, S. J. Young, The estimation of stochastic context-free grammars using the inside-outside algorithm, Computer speech & language 4 (1) (1990) 35–56.
[70] T. M. Cover, J. A. Thomas, Elements of Information Theory, John Wiley & Sons, 1991.
[71] A. Rosenberg, J. Hirschberg, V-measure: A conditional entropy-based external cluster evaluation measure., in: EMNLP-CoNLL, Vol. 7, 2007, pp. 410–420.
[72] C. Van Rijsbergen, Information retrieval., Butterworth, London, edition 2, 1979.
[73] L. R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proceedings of the IEEE 77 (2) (1989) 257–286.
[74] S. Abney, Partial parsing via finite-state cascades, Natural Language Engineering 2 (04) (1996) 337–344.
[75] D. H. Younger, Recognition and parsing of context-free languages in time n^3, Information and Control 10 (2) (1967) 189–208.
[76] K. R.M., Reducibility among combinatorial problems, in: R. Miller, J. Thatcher, J. Bohlinger (Eds.), Complexity of Computer Computations, The IBM Research Symposia Series. Springer, 1972, pp. 75–103.
[77] E. Danna, E. Rothberg, C. Le Pape, Exploring relaxation induced neighborhoods to improve MIP solutions, Mathematical Programming 102 (1) (2005) 71–90.
[78] F. Glover, Tabu search—part ii, ORSA Journal on computing 2 (1) (1990) 4–32.
[79] J.-B. Alayrac, P. Bojanowski, N. Agrawal, J. Sivic, I. Laptev, S. Lacoste-Julien, Unsupervised learning from narrated instruction videos, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp.

4575–4583.
[80] B. Burchfiel, G. Konidaris, Generalized 3D object representations using Bayesian eigenobjects, in: Proc. RSS, 2017.
[81] B. Bennett, H. Du, L. Gomez Alvarez, A. G. Cohn, Defining relations: a general incremental approach with spatial temporal case studies, in: Frontiers in Artificial Intelligence and Applications, Vol. 263, IOS Press, 2016, pp. 23–36.