Leveraging Text-to-Scene Generation for Language Elicitation and Documentation

Morgan Ulinski

Submitted in partial fulfillment of the

requirements for the degree

of Doctor of Philosophy

in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2019

ABSTRACT

Leveraging Text-to-Scene Generation for Language Elicitation and Documentation

Morgan Ulinski

Text-to-scene generation systems take input in the form of a natural language text and output a 3D scene illustrating the meaning of that text. A major benefit of text-to-scene generation is that it allows users to create custom 3D scenes without requiring them to have a background in 3D graphics or knowledge of specialized software packages. This contributes to making text-to-scene useful in scenarios from creative applications to education. The primary goal of this thesis is to explore how we can use text-to-scene generation in a new way: as a tool to facilitate the elicitation and formal documentation of language. In particular, we use text-to-scene generation (a) to assist field linguists studying endangered languages; (b) to provide a cross-linguistic framework for formally modeling spatial language; and (c) to collect language data using crowdsourcing. As a side effect of these goals, we also explore the problem of multilingual text-to-scene generation, that is, systems for generating 3D scenes from languages other than English.

The contributions of this thesis are the following. First, we develop a novel tool suite (the WordsEye Linguistics Tools, or WELT) that uses the WordsEye text-to-scene system to assist field linguists with eliciting and documenting endangered languages. WELT allows linguists to create custom elicitation materials and to document semantics in a formal way. We test WELT with two endangered languages, Nahuatl and Arrernte. Second, we explore the question of how to learn a syntactic parser for WELT. We show that an incremental learning method using a small number of annotated dependency structures can produce reasonably accurate results. We demonstrate that using a parser trained in this way can significantly decrease the time it takes an annotator to label a new sentence with dependency information. Third, we develop a framework that generates 3D scenes from spatial and graphical semantic primitives. We incorporate this system into the WELT tools for creating custom elicitation materials, allowing users to directly manipulate the underlying semantics of a generated scene. Fourth, we introduce a deep semantic representation of spatial relations and use this to create a new resource, SpatialNet, which formally declares the lexical semantics of spatial relations for a language. We demonstrate how SpatialNet can be used to support multilingual text-to-scene generation. Finally, we show how WordsEye and the semantic resources it provides can be used to facilitate elicitation of language using crowdsourcing.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

1	first person
3	third person
ABL	ablative
ACC	accusative
AFTER	'after'-ative
ALL	allative
AS.WELL	'as well, too, again, still'
ASP	aspect
ASSOC	associative
ASSOC.MOT	associative motion
AVER	aversive
BUT	'now consider this one; on the other hand, by contrast; but'
CAUS	causative
COM	comitative
CONT	'do continuously (non-motional)'
DAT	dative
DO.COMING	'do verb action while coming'
DO.PAST	'do verb action while moving past or through a place'
DS	different subject
DU	dual
ENG.TR	marker of English transitive verb
ERG	ergative
FACT	'it's a fact that'
FOC	focus
FREQ	frequentative
HAB	habitual involvement
INCH	inchoative
INS	instrumental
LOC	locative
NMLZ	nominalizer
NOM	nominative
NPC	non-past completive
NPP	non-past progressive
NUM	number
ONLY	'only, exclusively'
PC	past completive
PL	plural
POSS	possessive
PROP	proprietive
PST	past

PURP	purposive	SEMBL	semblative
RDP	reduplication	SG	singular
REFL	reflexive	SS	same subject
REL	relative	TMP.NOM	temporal nominal formative
REM.P.HAB	remote past habitual	TNS	tense

Acknowledgments

First, I would like to thank to my faculty advisor, Julia Hirschberg. I was incredibly lucky to have her as my advisor, and I would not have been able to accomplish as much as I have without her continuous support and encouragement. I am also grateful to Owen Rambow for his guidance and mentorship on many parts of the research that went into this thesis. Thanks also to Claire Cardie, Michael Riley, and John Hale, for their mentorship during my time as an undergraduate, and for providing me with my first research opportunities in natural language processing and computational linguistics.

Thank you to Mark Dras for giving me my first opportunity to work on Arrernte, including a trip to Alice Springs in Australia to work directly with Arrernte speakers. I also thank Myfany Turpin for sharing her knowledge and expertise on Arrernte and other Australian languages, both during that trip and after, as we began our work on the WordsEye Linguistics Tools. Thanks also to Daniel Kaufman for two semesters of his language documentation and field methods class, which provided me with a much stronger background on field linguistics and endangered languages.

I am grateful to all the members (official and honorary) of the Speech Lab for the supportive and welcoming environment I enjoyed during my time at Columbia: Bob Coyne, Fadi Biadsy, Rivka Levitan, Erica Cooper, Daniel Bauer, Arthi Ramachandran, Victor Soto, Sarah Ita Levitan, Nishi Cestero, Rose Sloan, and Zixiaofan (Brenda) Yang.

I am especially grateful to Bob Coyne, without whom this thesis would not have been possible, for his expertise and continual support on all things related to WordsEye. I would also like to thank the many project students who have helped with my work, including: Anusha Balakrishnan, Inna Fetissova, Alexandra Orth, Kevin Kwong, Carly Jackson, Lauren DeLucia, Eden Dolev, and Hannah Ahn.

Finally, I am tremendously grateful to my family for all their encouragement, help, and support over the years. My parents, my sisters, and my husband – none of this would have been possible without you.

Chapter 1

Introduction

Language elicitation is fundamental to many fields of study. Speech pathologists elicit language samples in order to diagnose and treat speech and language problems [Rvachew and Brosseau-Lapré, 2012]. Field linguists elicit language to help preserve endangered languages [Crowley and Thieberger, 2007]. Machine learning scientists elicit language they can use to train classifiers for machine translation [Zaidan and Callison-Burch, 2011], question answering [Chandu *et al.*, 2018], and other tasks [Wang *et al.*, 2012]. Teachers elicit language from students as a regular part of second language instruction [Rosenberg, 2009]. This thesis focuses on eliciting language in the context of both field linguistics and computational linguistics. *Language documentation* may include recording speech and text for archival purposes, annotating or translating language data, writing detailed descriptions of grammars, or creating formal computational models of language [Gippert *et al.*, 2006; Grenoble and Furbee, 2010]. In this thesis, we will address all of these, but our main interest is in what we call *formal documentation*, by which we mean linguistic grammars that describe a language's abstract system of structures and rules in a way that is both understandable to a linguist reading the grammar and that can also be processed using computational methods. Although the problem of language documentation is often associated primarily with field linguistics and its relevance to endangered language preservation, we believe that any computational model that is also human-readable is a form of language documentation. This includes many existing computational resources for high-resource languages, including

FrameNet [Ruppenhofer *et al.*, 2016] and WordNet [Fellbaum and Miller, 1998].

The primary goal of this thesis is to explore how we can use *text-to-scene generation* as a tool to facilitate the elicitation and formal documentation of language. Text-to-scene generation systems take input in the form of a natural language text and output a 3D scene illustrating the meaning of that text. A major benefit of text-to-scene generation is that it allows users to create custom 3D scenes without requiring them to have a background in 3D graphics or knowledge of specialized software packages. This contributes to making text-to-scene useful in scenarios from creative applications to education. In this thesis, we use text-to-scene generation (a) to assist field linguists studying endangered languages; (b) to provide a cross-linguistic framework for formally modeling spatial language; and (c) to collect language data using crowdsourcing. As a side effect of these goals, we also explore the problem of *multilingual* text-to-scene generation, that is, systems for generating 3D scenes from languages other than English.

1.1 Motivation

1.1.1 Endangered Language Preservation

Languages have appeared and disappeared throughout history. Today, however, languages are facing extinction at an unprecedented pace. Over 40 percent of the world's approximate 7,000 languages are at risk of disappearing. When languages die out, we not only lose the ability to study them linguistically, but we also lose access to an invaluable resource for studying the culture, history, and experience of peoples around the world [Endangered Languages Project]. Efforts to document languages and develop tools to support these efforts become even more important with the increasing rate of extinction. Bird [2009] emphasizes the particular need to make use of computational linguistics during fieldwork. This thesis addresses this need by applying text-to-scene generation to linguistic fieldwork and the study of endangered languages. Because 3D scenes naturally lend themselves to the elicitation of spatial and other graphical relations, we focus in particular on methods for the elicitation and documentation of spatial language.

1.1.2 Cross-linguistic Modeling of Spatial Language

Spatial relations have been studied in linguistics for many years. One study for English by Herskovits [1986] catalogs fine-grained distinctions in the interpretation of various prepositions. For example, she distinguishes among the uses of *on* to mean 'on the top of a horizontal surface' (*the cup is on the table*) or 'affixed to a vertical surface' (*the picture is on the wall*). Likewise, Feist and Gentner [1998] describe user perception experiments that show that the shape, function, and animacy of the FIGURE and GROUND are all factors in the perception of spatial relations as the lexical item *in* or *on*.

The relevance of object properties is language dependent. Bowerman and Choi [2003] describe how Korean linguistically differentiates between the lexical items *nehta*, which indicates putting something in a loose-fitting container (for example, fruit in a bag), versus *kkita*, which indicates putting something in a tight fitting wrapper (for example, hand in glove). Other languages, including English, do not make this distinction. Levinson [2003] and colleagues have also catalogued profound differences in the ways different languages encode relations between objects in the world. In particular, the Australian language Guugu Yimithirr and the Mayan language Tzeltal use absolute frames of reference to refer to the relative positions of objects. In Guugu Yimithirr, one can locate a chair relative to a table only in terms of cardinal points: saying, for example, that the chair is north of the table. In English such expressions are reserved for geographical contexts (for example, *Seattle is north of Portland*) and are never used for relations at what Levinson terms the "domestic scale". In Guugu Yimithirr one has no choice, and there are no direct translations for English expressions such as *The chair is in front of the table.* These kinds of distinctions, both within a language and between different languages, make the elicitation and documentation of spatial language an interesting and important problem in linguistics and computer science. In particular, tools for eliciting and subsequently documenting this kind of linguistic information should be an important part of endangered language preservation as a whole.

Formal, cross-linguistic models of spatial language are important not only for the purpose of documenting and preserving a diverse set of endangered languages, but also in many natural language processing applications for higher resource languages. Consider the following examples of machine translation. The prepositions are marked in boldface. The English sentence is a word-for-word gloss of the German sentence except for the preposition.

In our first example, English *on* is correctly translated to German *an*:[1]

(1) a. The painting is **on** the wall.

b. Correct translation: Das Gemälde ist **an** der Mauer/Wand.

c. Google Translate/Bing Translator (correct): Das Gemälde ist **an** der Wand.

However, the correct translation changes if we are relating a cat to a wall:

(2) a. The cat is **on** the wall.

b. Correct translation: Die Katze ist **auf** der Mauer.

c. Google Translate (incorrect): Die Katze ist **an** der Wand.

d. Bing Translator (incorrect): Die Katze steht **an** der Wand.

The problem here is that the English preposition *on* describes two different spatial configurations: 'affixed to', in the case of the painting, and 'on top of', in the case of the cat.[2]

Similar problems appear when we translate from German to English. The painting again translates correctly:

(3) a. Das Gemälde ist **an** der Mauer.

b. Correct translation: The painting is **on** the wall.

c. Google Translate/Bing Translator (correct): The painting is **on** the wall.

But when we replace the painting with the house, we no longer obtain the correct translation:

[1]Note that English *wall* should be translated to *Wand* if it is a wall which has a ceiling attached to it, and *Mauer* if it is freestanding and does not help create an enclosed three-dimensional space. We ignore this particular issue in this discussion.

[2]We set aside the interpretation in which the cat is affixed to the wall similarly to a clock, which is an extraordinary interpretation and would require additional description in either language.

(4) a. Das Haus ist **an** der Mauer.

b. Correct translation: The house is **at** the wall.

c. Google Translate/Bing Translator (incorrect): The house is **on** the wall.

The problem is again that the German preposition *an* corresponds to two different spatial configurations, ‘affixed to’ (painting) and ‘at/near’ (house).

In this thesis, we will explore how we can use text-to-scene technology to facilitate the formal modeling of cross-linguistic differences in the expression of spatial language. These models will be useful in improving machine translation and other multilingual NLP applications.

1.2 Tools and Resources

One of the main innovations of this thesis is that we combine existing tools and resources in different ways in order apply them to a new problem. While our focus is on how text-to-scene generation can be applied as a tool to facilitate language elicitation and documentation, our work incorporates many other kinds of tools as well. Some of the tools and resources we use are listed here:

- The **WordsEye Text-to-Scene System** [Coyne and Sproat, 2001]: to generate custom elicitation materials in Chapter 3; as the foundation of the semantics-to-scene system in Chapter 5; for graphical realization of spatio-graphic primitives in Chapter 6; as the source of images to elicit emotional language in Chapter 7.

- **VigNet** [Coyne *et al.*, 2011a]: to model lexical semantics in Chapter 3; as the initial source for the graphical primitives and ontology in Chapters 5 and 6; to help elicit imaginative sentences in Chapter 7.

- **XFST** [Beesley and Karttunen, 2003] and **XLE** [Crouch *et al.*, 2011]: to create computational models of Arrernte morphology and syntax in Chapter 3.

- **FieldWorks Language Explorer** [SIL International]: to annotate Arrernte sentences related to spatial language and case in Chapter 3.

- The **Max Planck topological relations picture series** [Bowerman and Pederson, 1992]: as the model for the topological relations scenes used to elicit Nahuatl in Chapter 3; as a reference for expanding the set of spatio-graphic primitives in Chapter 6.
- The **Picture Series for Positional Verbs** [Ameka *et al.*, 1999] and the **Motion Verb Stimulus Kit** [Levinson, 2001]: to obtain sentences for the corpus/treebank introduced in Chapter 4; as a reference for expanding the set of spatio-graphic primitives in Chapter 6.
- **Tree Editor (TrEd)** [Pajas and Štěpánek, 2008]: to annotate our new corpus with dependency information in Chapter 4; as a component of the UI for the annotation experiments in Chapter 4.
- **MSTParser** [McDonald *et al.*, 2006] and **MaltParser** [Nivre *et al.*, 2006]: to train parsers on incrementally increasing amounts of data in Chapter 4.
- **FrameNet** [Ruppenhofer *et al.*, 2016]: as a prototype for and source of some of our spatial frames in Chapter 6.
- The **Princeton WordNet of English** [Princeton University], **GermaNet** [Henrich and Hinrichs, 2010; Hamp and Feldweg, 1997], **Open German WordNet** [Siegel], and the **EuroWordNet Interlingual Index (ILI)** [Vossen, 1998]: to create the German lexical mapping in Chapter 6.
- The **Stanford CoreNLP Toolkit** [Manning *et al.*, 2014]: to perform German and English morphological and syntactic parsing in the SpatialNet text-to-scene pipeline described in Chapter 6.
- **Amazon Mechanical Turk** [Amazon Mechanical Turk, Inc., 2018]: as the platform for the crowdsourcing examples in Chapter 7.

1.3 Outline of Thesis

Throughout this thesis, we use the WordsEye text-to-scene system [Coyne and Sproat, 2001; Coyne, 2017], which converts English text into 3D scenes representing the meaning of that text. Our goal is to use 3D scenes created in WordsEye to facilitate the elicitation of language data; subsequently, the text-to-scene system provides a natural semantic grounding for formally documenting a language's lexical semantics. We apply this methodology first to the problem of eliciting and documenting endangered languages, with the development of the WordsEye Linguistics Tools, or WELT. WELT has two modes of operation, the first for elicitation and the second for documentation. The first mode of operation, which we call WELT English, provides tools to assist with the elicitation of language data. We have developed user interfaces that allow users to create custom elicitation materials in the form of 3D scenes and to build and conduct elicitation sessions based around those scenes. We have also created a prototype of the second mode of operation, which we call WELT L2. The purpose of WELT L2 is to provide a way to formally document the semantics of the endangered language. Formal hypotheses specified using WELT L2 can be verified using a text-to-scene system that takes input in the endangered language, analyzes it based on the formal model, and generates a picture representing the meaning.

We have tested WELT English by creating a set of scenes representing basic topological relations and eliciting descriptions from a native speaker of Nahuatl, an endangered language spoken in Mexico. WELT English has also been used by Anusha Balakrishnan to elicit Ikota, an endangered language spoken along the border of Gabon and the Republic of the Congo. In addition, many of the scenes we used to elicit Nahuatl were created by undergraduate linguistics students who did not have a computational background. These facts demonstrate that although the WELT elicitation tools have not yet been deployed broadly, they are reusable by different users and for different languages. Our prototype of WELT L2, including converting endangered language text into a 3D scene, works with a limited grammar of Arrernte, an Australian aboriginal language spoken in Alice Springs. The goal of our work on documenting Arrernte with WELT L2 is to show that the success we have had using WELT English with different endangered languages can be extended to the documentation tools. In Chapter 3, we describe the WordsEye Linguistics Tools and demonstrate their use with two endangered languages: Nahuatl and Arrernte.

In order to make full use of the WELT workflow, including verification of hypotheses via text-to-scene generation, it is necessary for the system to be able to parse input text into its morphology and syntax in addition to semantics. SIL FieldWorks Language Explorer (FLEx) [SIL International] provides the means to create a morphological parser without knowledge of any particular grammatical formalism, by training the parser in the background as the linguist annotates their language data. The ability to easily document syntax, however, is largely missing from existing documentation tools. In Chapter 4, we explore the question of whether a tool similar to FLEx might potentially be created for syntax. We introduce and describe a new multilingual treebank we have created, consisting of short descriptions of spatial configurations and motion events, annotated with syntactic dependencies and other linguistic information. We use this treebank to perform experiments in which we train a dependency parser on incrementally increasing small amounts of data. This simulates the scenario of a parser being learned in the background while a linguist gradually annotates more sentences. We also conduct experiments that demonstrate that using a parser trained in this manner improves the performance of human annotators annotating dependency structures. Our experiments show that it is possible to extend FieldWorks' approach to learning a morphological parser into the realm of syntax.

Generating scenes from natural language simplifies the creation of custom 3D content, but there are situations when a natural language interface is not ideal. For example, 3D scenes created from natural language are limited by any errors the system makes processing language of the input text. Even in the absence of system errors, it can be difficult to achieve a precise spatial configuration due to the ambiguity of natural language. In addition, using English to create scenes that will be used for elicitation in another language may introduce unwanted biases and assumptions about the target language. We address these problems by adapting WordsEye into a system that can generate a 3D scene from semantic primitives representing basic spatial relations and graphical properties. This gives users direct access to the underlying semantics of 3D scenes being created instead of relying on black-box conversion from input text to rendered scene. In Chapter 5, we describe the architecture of the system and the incorporation of this functionality into the WELT elicitation tools.

One issue with WELT is that, in order to verify formal models using a text-to-scene workflow, the semantic documentation must ultimately map into VigNet, which, having been developed specifically for English, is often biased both to the English language and toward Western culture. Our adaptation of WordsEye to allow scenes to be generated from a language-independent semantic representation enables us to avoid our dependence on the English-specific components of WordsEye, while still maintaining the benefit of being able to create a text-to-scene system for a language. This adaptation also provides the means to formally document spatial semantics using a *language-independent* semantic grounding. In Chapter 6, we describe how we use this semantic grounding as part of a new multilingual resource, SpatialNet, which provides a framework for defining the lexical semantics of spatial relations in a language. SpatialNet uses a declarative format to link linguistic expressions both to semantic frames and to actual spatial configurations. Because SpatialNet provides a link between surface language and semantic primitives, it can also be used in conjunction with our modified WordsEye system to generate 3D scenes from text.

Most of the discussion of language elicitation in this thesis focuses on using text-to-scene generation in the context of endangered languages, with a trained field linguist eliciting language directly from native speaker informant. However, text-to-scene generation can also be useful to facilitate elicitation of language data in other contexts. In Chapter 7, we show three examples of how WordsEye can be used in conjunction with crowdsourcing to elicit different types of language data. First, we use 3D scenes previously created by WordsEye users and published to the WordsEye gallery to create a corpus of descriptions containing emotional language. Second, we show how generating 3D scenes from spatial and graphical primitives can be used with crowdsourcing to elicit targeted descriptions of spatial configurations. Finally, we use the lexical and semantic information contained in VigNet to obtain a set of descriptions of imaginative scenarios.

In Chapter 8, we conclude by summarizing the contributions of this thesis and discussing future work.

Chapter 2

Background on WordsEye and VigNet

2.1 Introduction

WordsEye [Coyne and Sproat, 2001; Coyne, 2017] is a system for automatically converting (English) natural language text into 3D scenes representing the meaning of that text. To do this, WordsEye relies on VigNet [Coyne *et al.*, 2011a], a knowledge base of lexical, semantic, and graphical information. We will describe VigNet in more detail in Section 2.4.

WordsEye supports language-based control of spatial relations, textures and colors, collections, facial expressions, and poses. It handles simple anaphora and co-reference resolution, allowing for a variety of ways of referring to objects. The system assembles complex scenes from a library of approximately 3,000 3D objects and 10,000 2D images tied to a lexicon of 20,000 nouns. These include a wide variety of common objects (including variations of the same basic type, such as different types of doors or chairs) and textures (e.g. wood, grass, granite). WordsEye also supports a range of graphical primitives and properties that are used for spatial relations, (different senses of *in*, *on*, lateral relations, etc.), spatial properties (absolute and relative sizes and aspect ratios), and surface properties (colors, opacity, reflectivity, etc.). These primitives in conjunction with the objects and semantic knowledge about those objects (such as defaults for size, orientation, and top surface regions) allow the scene to be composed.

WordsEye is a good choice as a tool for our work for several reasons. First, it is now a reliable web application[1] that has been online since November 2015, with approximately 35,000 registered real-world users. Before the launch of the web application, earlier versions of WordsEye were used for many years on a smaller scale. For example, an earlier online version of WordsEye (then known as Semantic Light) had around 3,000 registered users from 2004–2007. Another version of WordsEye was used in 2010 as an educational tool, helping students develop language skills during a summer literacy enrichment program run by the Harlem Educational Activities Fund [Coyne *et al.*, 2011b]. This history of real-world usage means that WordsEye will be a more stable tool than other text-to-scene systems that have been developed (we will briefly mention some of these in Section 2.2). Second, WordsEye supports a wide variety of spatial relations, graphical properties, and 3D objects. This broad coverage allows for flexibility in the kinds of 3D scenes that can be created. In addition, the WordsEye system works by passing input text through a pipeline with distinct modules for different stages of language processing and graphical analysis. This allows us to more easily reuse parts of the system while replacing others with new interfaces.

In this thesis, we will discuss using WordsEye in the following ways:

- In Chapter 3, we describe how we are using WordsEye to create a comprehensive tool for field linguistics, to be used (a) to elicit descriptive language from native speakers of endangered languages and (b) to serve as a test-bed for syntactic/semantic grammars of endangered languages.

- In Chapter 5, we describe using WordsEye to create a new system that can generate a 3D scene from semantic primitives representing basic spatial and graphical relations.

- In Chapter 6, we describe using WordsEye as a realization engine for a deep semantic representation of spatial relations. We also show how this can be applied to support multilingual text-to-scene-generation.

- Finally, in Chapter 7, we show how WordsEye can be used to help with collecting

[1]`http://www.wordseye.com`

language data using crowdsourcing.

In this chapter, we describe WordsEye and VigNet in some detail, which will help clarify how the different parts of the system are used when we discuss them later in this thesis. Section 2.2 discusses related work on text-to-scene generation. Section 2.3 provides an overview of the WordsEye architecture and processing pipeline. Section 2.4 describes VigNet. We conclude in Section 2.5.

2.2 Related Work

Several systems exist for producing graphics from natural language sources. Adorni *et al.* [1984], Simmons [1975], and the Put system [Clay and Wilhelms, 1996] were early systems used to position objects in a 3D space. Badler *et al.* [2000] and AVis [O'Kane *et al.*, 2004] used language to control animated characters in a closed virtual environment. CarSim [Dupuy *et al.*, 2001] is a domain-specific systems to create animations from natural language descriptions of accident reports. Glass [2008] describe a system for transforming text sourced from popular fiction into corresponding 3D animations without prior language simplification. 3SVD [Zeng *et al.*, 2005] is a 3D scene creation system using story-based descriptions. Parisi *et al.* [2007] describe an ontology-driven generation of 3D animations for training and maintenance. CONFUCIUS [Ma, 2006] is a multimodal text-to-animation system that generates animations of virtual humans from single sentences containing an action verb. A survey of these and other text-to-graphics systems is given in Hassani and Lee [2016]. In most existing systems, the referenced objects, attributes, and actions are typically relatively small in number or targeted to specific pre-existing domains. WordsEye was one of the first text-to-scene systems designed for general use rather than a specific domain.

Another recent system, introduced in Chang *et al.* [2014a], focuses on indoor scenes and supports a large number of 3D objects; however, it supports fewer spatial relations and graphical properties than WordsEye. The work described in Chang [2015] also emphasizes incorporating real-world knowledge in order to infer additional information not directly specified in the input text. WordsEye, on the other hand, creates a 3D scene that is a more

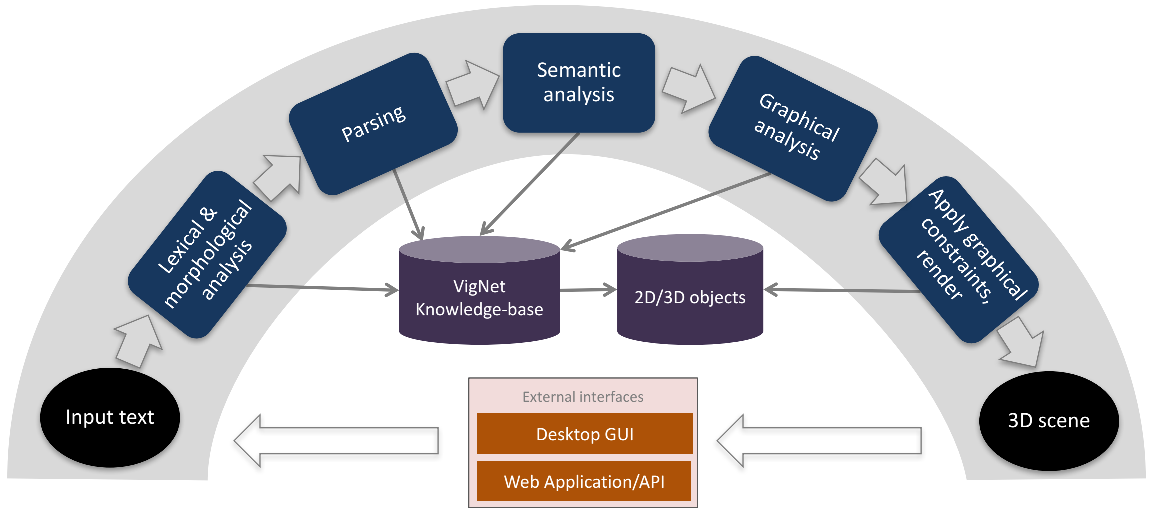


Figure 2.1: WordsEye System Architecture

precise representation of the meaning conveyed by the input text. In addition, a major focus of the current work on WordsEye is in developing a robust, stable application for real-world users. This makes it a good choice for our research, which requires a text-to-scene system to be a reliable foundation for us to build upon.

2.3 The WordsEye Text-to-Scene System

In this section, we provide an overview of how the WordsEye system works. We explain how WordsEye converts its input from text to semantics to graphical primitives and finally to a rendered 3D scene. This will help clarify how its graphical-semantic primitives, lexical dictionary, knowledge-base, and semantic formalisms are utilized in the tools described in this thesis.

The architecture of WordsEye is shown in Figure 2.1. Input text is converted into a 3D scene by a series of processing stages. Throughout the process, the system relies on VigNet, a knowledge base of lexical, semantic, and graphical information. VigNet includes an ontology of semantic types (including both abstract concepts and specific 3D objects), a hierarchy of relations, and a set of assertions that represent real-world knowledge by applying relations to concepts. It also includes links between the concepts/relations and English lexical items. We will describe VigNet in more detail in Section 2.4.

The system operates by first tokenizing each input sentence into lexical items (including modifiers like contractions or possessives) and possible parts-of-speech. It does this by identifying words in its lexicon, which consists of about 30,000 English lexical items. For each lexical item, the dictionary includes information such as verbal inflections, alternative spellings, and singular/plural noun forms. WordsEye tokenizes the text using custom heuristics; tokens are then labeled with (possibly multiple) part of speech tags found in the lexical dictionary. Multi-word prepositions (e.g. *to the left of*) are also identified at this stage. Then text is then parsed into a constituent tree using a custom CYK [Younger, 1967] grammar of approximately 1000 rules. The grammar identifies the head constituent at each level of the parse, allowing the automatic construction of a syntactic dependency tree from the constituency parse. WordsEye's grammar has been optimized to handle descriptions of spatial and graphical configurations. It usually handles low-level lexical-semantic phenomena, such as adverbial modifiers on spatial prepositions, better than off-the-shelf parsers. For example, the English dependency parser in the Stanford CoreNLP Toolkit [Manning *et al.*, 2014] gives incorrect analyses on simple sentences like *the dog jumped 3 feet over the table*, interpreting *3 feet* as a direct object of *jump* (in this case, the noun phrase *3 feet* is acting as an adverbial modifier of the spatial preposition, not as an object of the verb).

After converting the constituent parse into a labeled syntactic dependency structure, the dependency structure is processed for anaphora and other co-reference, which is especially important for depicting multi-sentence input. These resolved structures are converted to lexical-semantic relations using lexical valence patterns and other lexical and semantic information. In this process a lexical item like *on* is converted to a specific graphical primitive relation. For example, in *the picture is on the wall*, *on* is converted to a graphical primitive that puts the object on the front surface of the wall, whereas in *the vase is on the table*, *on* is converted a different primitive that puts the vase on the top surface of the table. The semantic decomposition rules examine the VigNet knowledge base to determine the properties of the objects and pick the correct spatial decomposition. In this case, a wall is known to primarily be a vertical surface, while a table is a horizontal surface.

For higher-level semantic constructs, like verbs and their arguments, the semantics can first be converted into a *vignette*, which is a particular real-world realization of a semantic meaning. For example, the verb *wash* can be realized in many different ways, depending on whether one is washing dishes versus washing one's hair versus washing a car. Vignettes are intermediate graphically-inspired interpretations and representations that are in turn decomposed into low-level graphical primitives. We discuss vignettes in more detail in Section 2.4.

Next, WordsEye chooses objects to satisfy the input semantics. For example, if the word *dog* or concept DOG.N was specified, it chooses a particular 3D model of a dog to match the given concept. This is primarily a random selection, although certain higher-quality 3D models are given preference over others. After assigning objects, WordsEye performs graphical analysis to compose the scene. The lexical-semantic relations are converted to a *graphical semantics*, which is a set of graphical constraints representing the position, orientation, size, color, texture, cardinality, and poses of objects in the scene. This graphical semantics can be thought of as a semantic grounding; it is used to construct and render a 3D scene.

Construction of the final graphical semantics involves supplying additional graphical constraints to impose some real-world defaults. For example, scenes are often under-constrained. Normally one expects objects to be on the ground versus floating in the air, but usually the ground is not specifically mentioned in the input. In addition, objects generally do not occupy the same location; for a phrase like *the cat and the dog*, the objects should be next to each other rather than inter-penetrating. WordsEye infers these additional constraints as well as associated objects such as a ground and sky. A similar process occurs with the surface properties of objects (such as colors, textures, and reflectivity) and with light sources. The system then applies the entire set of constraints to position the objects in the scene, and renders the final scene.

A Note on WordsEye External Interfaces. There are two main interfaces to Words-Eye which will be used in this thesis: an older desktop application that runs on Mac OS X, and the latest version of WordsEye, which is a web application that can also be accessed through a web API. We use the desktop application for the development of the WordsEye Linguistics Tools (Chapter 3); we use the web API for generating scenes from semantics primitives (Chapter 5), and as the realization engine for SpatialNet (Chapter 6). For the crowdsourcing experiments in Chapter 7, we used the web application accessed both through a web browser and the API. Although the basic functionality of the two interfaces to WordsEye is the same, there are some differences. In general, the newer web application provides more functionality, although one notable exception is that the desktop application supports putting characters in different poses, while the web application does not.

2.4 VigNet

To interpret input text, WordsEye uses a lexical resource called VigNet [Coyne *et al.*, 2011a]. VigNet is inspired by and based on FrameNet [Baker *et al.*, 1998; Ruppenhofer *et al.*, 2016], a resource for lexical semantics. In FrameNet, lexical items are grouped together in frames according to their shared semantic structure. Every frame contains a number of frame elements (semantic roles) which are participants in this structure. The English FrameNet defines the mapping between syntax and semantics for a lexical item by first providing a mapping between the lexical item and frames which it can verbalize, and by providing then for each such frame lists of valence patterns that map syntactic functions to frame elements.

VigNet extends FrameNet in two ways in order to capture *graphical semantics*, the knowledge needed to generate graphical scenes from language. First, graphical semantics are added to the frames by adding primitive graphical (typically, spatial) relations between the frame element fillers. Second, VigNet distinguishes between meanings of words that are distinguished graphically. For example, the specific objects and spatial relations in the graphical semantics for *cook* depend on the object being cooked and on the culture in which it is being cooked (cooking turkey in Baltimore vs. cooking an egg in Alice Springs), even though at an abstract level *cook an egg in Alice Springs* and *cook a turkey in Baltimore* are perfectly compositional semantically. Frames augmented with graphical semantics are called *vignettes*.

The descriptions of the graphical semantics in vignettes make use of a set of object-

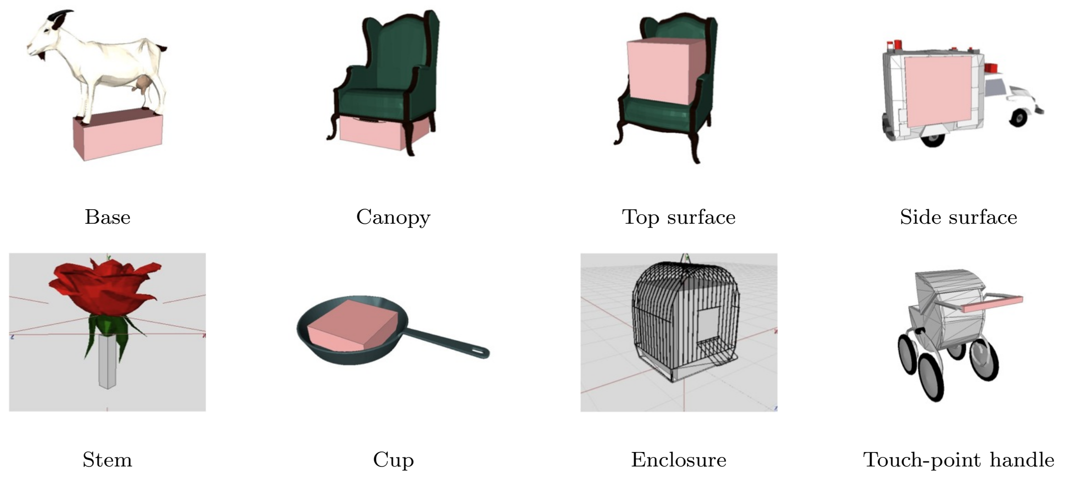


Table 2.1: Spatial affordances, represented by the boxes associated with each object, designate regions of those objects used in resolving spatial relations.

centric properties called *affordances* [Gibson, 1977; Norman, 1988]. The concept of affordances has a long history in the study of ergonomics and the psychological interpretation of the environment. WordsEye includes as affordances any functional or physical property of an object that allows it to participate in actions and relations with other objects. For example, a SEAT of a chair is used to support a sitter or small object and the INTERIOR of a box is used to hold the contents. WordsEye has a rich set of spatial affordances. Some examples of these are CUPPED REGIONS for objects to be *in*, CANOPIES for objects to be *under*, and TOP SURFACES for objects to be *on*. See Table 2.1 for more examples of spatial affordances. Affordances are particularly important for interpreting spatial prepositions, which are often ambiguous; for example *apple in the bowl* (CONTAINMENT) vs. *boat in water* (EMBEDDING)). To resolve the ambiguity, a constraint is placed on on the GROUND argument that demands that its filler offers an INTERIOR affordance.

Information about the 3D objects in WordsEye is organized in VigNet into an *ontology*. The ontology consists of semantic concepts that are linked together with ISA relations. The ontology supports multiple inheritance, allowing a given concept to be a sub-type of more than one concept. For example, a PRINCESS.N is a subtype of both FEMALE.N and ARISTOCRAT.N, and a BLACK-WIDOW.N is a subtype of both SPIDER.N and POISONOUS-ENTITY.N.

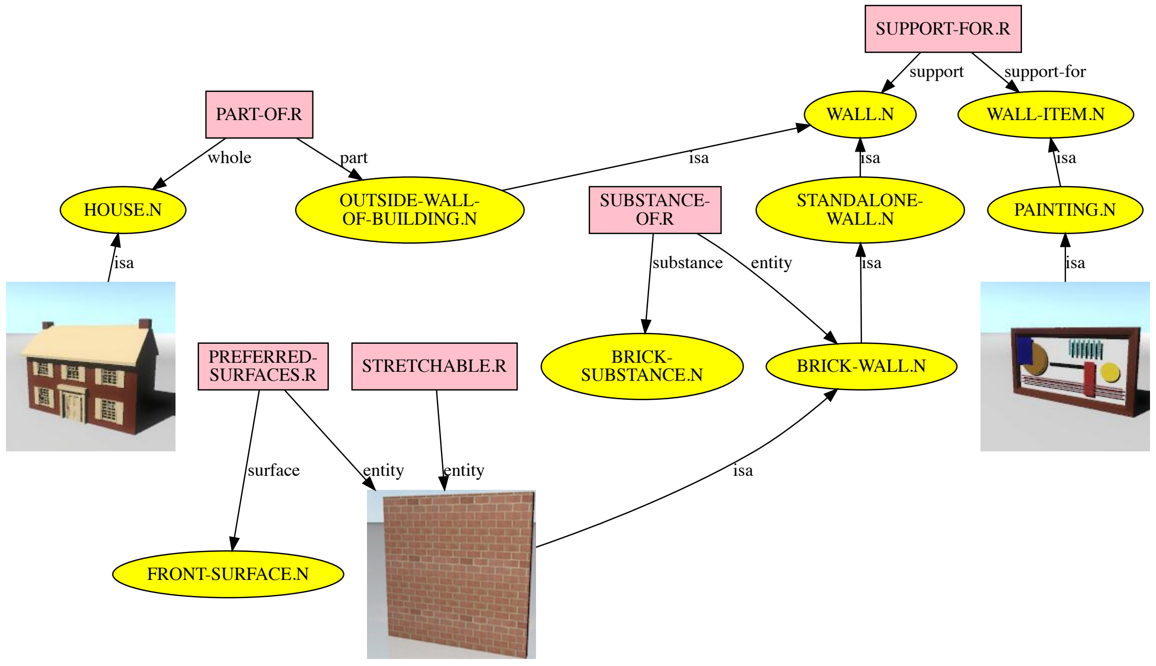


Figure 2.2: A portion of the WordsEye ontology. Semantic types are represented by yellow ovals. Assertions are represented by pink rectangles. The thumbnails represent particular 3D object models.

Types include both 3D objects and more general semantic concepts. For example, a particular 3D model of a dog is a subtype of the general DOG.N. Every 3D object has a semantic type and is inserted into the ontology. The 3D parts of 3D objects are also represented as types in the ontology. WordsEye also includes lexicalized concepts (e.g. *chair* tied to CHAIR.N) in the ontology. If a lexical item has more than one word sense, the different word senses are linked to different concepts. The semantic concepts in VigNet include the graphical objects available in WordsEye as well as concepts that are not currently supported in WordsEye. While WordsEye might only have a handful of graphical objects for dogs, VigNet has concepts representing all common types of dogs, even if there is no graphical object associated with them.

The ontology includes a knowledge base of assertions that provide more information about semantic concepts. Assertions include sizes of objects and concepts, their parts, their colors, what they typically contain, what affordances they have, the typical location-

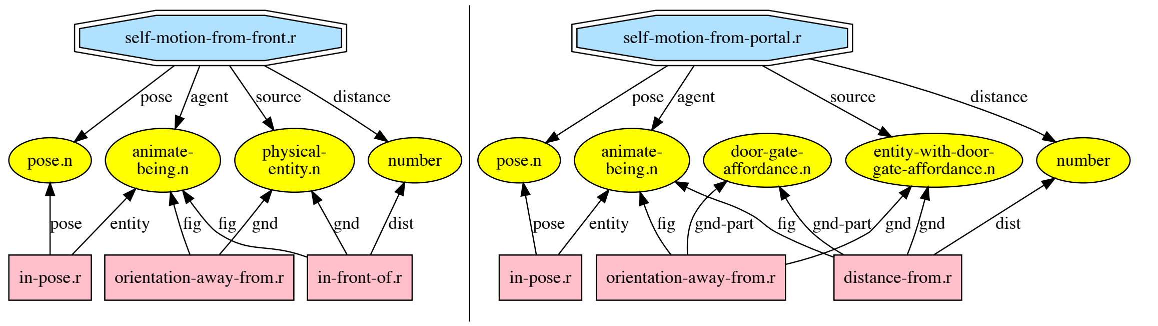


Figure 2.3: Two frames augmented with primitive graphical relations. The high-level semantics of SELF-MOTION-FROM-FRONT.R and SELF-MOTION-FROM-PORTAL are decomposed into different sets of objects and primitive graphical relations. Frames are represented by blue octagons, yellow ovals represent semantic constraints, and primitive graphical relations are represented by pink rectangles.

s/habitats of objects, and information about their function. Spatial affordances and other properties can be applied to both 3D graphical objects and to more general semantic types. For example, the general semantic type CUP.N has a CUPPED REGION affordance, since this affordance is shared by all cups. A particular 3D graphical object of a cup might have a HANDLE affordance, while another might have a LID affordance, but these spatial affordances are not tied to the super-type CUP.N. Figure 2.2 shows a small subset of the semantic hierarchy along with examples of assertions.

Figure 2.3 shows an example of two vignettes: SELF-MOTION-FROM-FRONT.R and SELF-MOTION-FROM-PORTAL.R. Both are sub-types of SELF-MOTION-FROM.R. The yellow ovals contain semantic constraints on the objects used to instantiate the frame. For example, while the relation SELF-MOTION-FROM-FRONT.R requires only that the source of the motion be a PHYSICAL-ENTITY.N, SELF-MOTION-FROM-PORTAL.R requires that the source has a DOOR-GATE-AFFORDANCE.N as a part. The primitive graphical relation ORIENTATION-AWAY-FROM.R in the decomposition for the latter then also assigns the argument GND-PART to be the DOOR-GATE-AFFORDANCE in addition to assigning the GND argument to the source of the motion.

2.5 Conclusion

In this chapter, we have provided an overview of the WordsEye text-to-scene system. We will use WordsEye as a tool for much of the research presented in this thesis, as we explore how text-to-scene generation can be used in new areas and applications. The modularized design of the WordsEye architecture and processing pipeline is a major benefit to the work in this thesis that builds on the WordsEye system. It allows us to easily reuse some parts of the system while replacing others with customized components. For example, the diagram in Chapter 3, Figure 3.2 shows how we modify the WordsEye architecture to be used by the WordsEye Linguistics Tools. Similarly modified architecture diagrams are shown in Chapter 5, Figure 5.1, when we discuss modifying the WordsEye pipeline to allow for semantic input and output, and in Chapter 6, Figure 6.7, when we describe using SpatialNet for text-to-scene generation.

Chapter 3

The WordsEye Linguistics Tools (WELT): Using Graphics Generation in Linguistic Fieldwork

3.1 Introduction

Although languages have appeared and disappeared throughout history, today languages are facing extinction at an unprecedented pace. Over 40% of the estimated 7,000 languages in the world are at risk of disappearing. When languages die, we lose access to an invaluable resource for studying the culture, history, and experience of people who spoke them [Endangered Languages Project]. Efforts to document languages and develop tools to support these efforts become even more important with the increasing rate of extinction. Bird [2009] emphasizes a particular need to make use of computational linguistics during fieldwork.

In this chapter,[1] we describe how we have designed the WordsEye Linguistics Tools, or WELT,[2] to address this issue by helping field linguists study endangered languages. WELT

[1]Some of the material in this chapter was published in Ulinski *et al.* [2014a,b], based upon work supported by the National Science Foundation under Grant No. 1160700: “Using Computational Tools to Facilitate Corpus Collection and Language Use in Arrernte (aer).”

[2]In German, *Welt* means “world”.

is a novel tool for the elicitation and documentation of endangered languages. The purpose of WELT is to provide field linguists with a tool for eliciting endangered language data and formally documenting the semantics of a language. WELT has two modes of operation. The first mode of operation, which we call WELT English, consists of tools for elicitation. The second mode of operation, which we call WELT L2, consists of tools for documentation. We will demonstrate WELT's use on scenarios involving two endangered languages, Arrernte and Nahuatl.

WELT English provides tools for building and organizing elicitation sessions based on custom 3D scenes. English input automatically generates a picture using WordsEye which can then be used to elicit a description in the target language. The elicitation tools in WELT English provide several advantages over using a set of pre-fabricated static pictures like those commonly used by field linguists today. Users are not limited to a fixed set of pictures but may, instead, create and modify scenes in real time based on the informants' answers. This allows them to create additional follow-up scenes and questions on the fly. In addition, since the pictures are 3D scenes, the viewpoint can easily be changed, allowing exploration of linguistic descriptions based on different frames of reference. This is particularly useful in eliciting spatial descriptions. Finally, since WordsEye can also be extended to include custom 3D content, the user can customize the images used for elicitation to be maximally relevant to their informants. We have created user interfaces for WELT English which integrate the basic functionality of WordsEye with tools for organizing sets of scenes for elicitation sessions, and other tools for annotating and recording language data during the elicitation session. We have tested these tools by creating a set of scenes representing topological relations and eliciting descriptions from a native speaker of Nahuatl, an endangered language spoken in Mexico. WELT English has also been used by Anusha Balakrishnan to elicit Ikota, an endangered language spoken along the border of Gabon and the Republic of the Congo. This was done as her final project for a class on field methods and language documentation at Columbia University. In addition, many of the scenes we used to elicit Nahuatl were created by two undergraduate linguistics students who did not have a computational background. These facts demonstrate that although the WELT elicitation tools have not yet been deployed broadly, they are reusable by different users and for different languages.

WELT L2 provides a means to document the semantics of a language in a formal way. Although there are several existing tools that allow field linguists to formally document phonetics and morphology, most notably the SIL FieldWorks Language Explorer [SIL International], there is no such tool for the formal documentation of semantics. The WELT documentation tools are designed to address this gap. Formally documenting a language with WELT L2 also creates a text-to-scene system that takes input in the endangered language, analyzes it based on the formal model, and generates a picture representing the meaning. This text-to-scene system will allow linguists to test the theories they develop with native speakers, making changes to grammars and semantics in real time. We have created a prototype of WELT L2 which works with a limited grammar of Arrernte, an Australian aboriginal language spoken in Alice Springs. The goal of our work on documenting Arrernte with WELT L2 is to show that the success we have had with using WELT English with different endangered languages can be extended to the documentation tools.

In this chapter, we will describe our development of the two modes of operation, WELT English and WELT L2. We will discuss results from conducting elicitation sessions with a native speaker of Nahuatl, and from documenting the lexical semantics of Arrernte. We will also describe a basic Arrernte text-to-scene system created in WELT. In Section 3.2 we discuss prior work on computational tools for field linguistics. In Section 3.3, we provide more information about the two endangered languages discussed in this chapter, Arrernte and Nahuatl. In Section 3.4 we present an overview of the WELT system, including the modifications we made to WordsEye. In Section 3.5, we describe our development of a morphological and syntactic grammar for Arrernte. We describe using WELT English for elicitation in Section 3.6 and describe the WELT L2 tools for language documentation in Section 3.7, including a description of the system for generating 3D scenes from endangered language input. In Section 3.8, we describe our pilot work toward using WELT to study the relationship between case and semantic interpretation of a sentence in Arrernte. We conclude in Section 3.9.

3.2 Related Work

Computational tools for field linguistics fall into two categories: tools for native speakers to use directly, without substantial linguist intervention, and tools for field linguists to use. Tools intended for native speakers include the PAWS starter kit [Black and Black, 2009], which uses the answers to a series of guided questions to produce a draft of a grammar. Similarly, Bird and Chiang [2012] describe a simplified workflow and supporting MT software that lets native speakers produce useable documentation of their language on their own.

One of the most widely-used toolkits in the latter category is SIL FieldWorks [SIL International], or specifically, FieldWorks Language Explorer (FLEx). FLEx includes tools for eliciting and recording lexical information, dictionary development, creating interlinear glossed text, analysis of discourse features, and morphological analysis. An important part of FLEx is its "linguist-friendly" morphological parser [Black and Simons, 2008], which uses an underlying model of morphology familiar to linguists, is fully integrated into lexicon development and interlinear text analysis, and produces a human-readable grammar sketch as well as a machine-interpretable parser. The morphological parser is constructed "stealthily" in the background, and can help a linguist by predicting glosses for interlinear texts.

Several computational tools aim to simplify the formal documentation of syntax by eliminating the need to master particular grammar formalisms. First is the PAWS starter kit [Black and Black, 2012], a system that prompts linguists with a series of guided questions about the target language and uses their answers to produce a PC-PATR grammar [McConnel and Black, 2006]. The LinGO Grammar Matrix [Bender *et al.*, 2002] is a similar tool developed for HPSG that uses a type hierarchy to represent cross-linguistic generalizations. The LinGO Grammar Matrix [Bender *et al.*, 2002] facilitates formal modeling of syntax by generating basic HPSG "starter grammars" for languages from the answers to a typological questionnaire. Extending a grammar beyond the prototype, however, does require extensive knowledge of HPSG, making this tool more feasibly used by grammar engineers and computational linguists.

Linguist's Assistant [Beale and Allman, 2011] provides a corpus of semantic representations for linguists to use as a guide for elicitation. After eliciting the language data, a linguist writes rules translating these semantic representations into surface forms. The result is a description of the language that can be used to generate text from documents that have been converted into the semantic representation. Linguists are encouraged to collect their own elicitations and naturally occurring texts and translate them into the semantic representation.

For semantics, the most common resource for formal documentation across languages is FrameNet [Fillmore *et al.*, 2003]; FrameNets have been developed for many languages, including Spanish, Japanese, and Portuguese. Most start with English FrameNet and adapt it for the new language; a large portion of the frames end up being substantially the same across languages [Baker, 2008]. However, FrameNet is also targeted toward computational linguists. ParSem [Butt *et al.*, 2002] is a collaboration to develop parallel semantic representations across languages, by developing semantic structures based on LFG. Neither of these resources, however, are targeted at helping non-computational linguists formally document a language, as compared to the morphological parser in FLEx or the syntactic documentation in PAWS.

In general, we also lack tools for creating custom elicitation materials. With WELT, we hope to fill some of the gaps in the range of available field linguistics tools. WELT will enable the creation of custom elicitation material and facilitate the management sessions with an informant. WELT will also enable formal documentation of the semantics of a language without knowledge of specific computational formalisms. This is similar to the way FLEx allows linguists to create a formal model of morphology while also documenting the lexicon of a language and glossing interlinear texts.

3.3 Languages

We have used WELT with two endangered languages, Arrernte and Nahuatl. This section provides background information about these languages.

3.3.1 Arrernte

Arrernte is an endangered language of the Arandic language group, a group of languages spoken in Central Australia. The main branches of Arrernte can be broadly split into Eastern/Central and Western variants. We will focus on Eastern/Central Arrernte, for reasons of location and resources. Figures for the number of speakers of Arrernte vary quite widely, and are further complicated by questions of degree of fluency. The most common figures are 4500-6000 for speakers of the Arandic group as a whole, and about 1800 for speakers of Eastern/Central Arrernte [Aboriginal Art and Culture]. The Arrernte people live in Central Australia, in and around Alice Springs (*Mparntwe* in Arrernte). Arrernte has a number of characteristics quite unlike English or other Indo-European languages. These include extensive use of morphology, fundamentally free word order (but with word order preferences and restrictions on various sub-parts of the language), lack of a copula verb, and "quasi-inflections" on verbs including a "category of associated motion."

While some aspects of Arrernte are well documented [Wilkins, 1989; Henderson, 1998], others have presented greater challenges. In particular, a number of idiosyncratic lexical and morphological features of the language relating to spatial relations have not previously been given a complete description. These are linguistically interesting because they relate directly to how a language is used by its speakers to describe their perceived reality. In Arrernte, for example, (1) in the category of associated motion there are verb inflections for concepts such as DO ACTION X QUICKLY WHILE MOVING DOWNWARDS; (2) indication of location can be much more fine-grained than in English, differentiating whether a person or thing is at rest, or has been moved, or is being observed; and (3) directions are expressed using cardinal (compass) point terms rather than relative terms. Another topic that needs to be explored in Arrernte is the relationship between case and the semantic interpretation of the sentence; it is possible to significantly alter the meaning of a sentence by changing the case on one of the nouns. In a related Arandic language, Kaytetye, adding allative case to the direct object of a sentence with the predicate *ltare* 'shoot' changes the meaning from shooting and hitting the object to firing a shot and not actually hitting it. We will discuss how WELT might be employed to study Arrernte case in Section 3.8.

In terms of linguistic analysis of Eastern Arrernte, there is good coverage of the grammar [Strehlow, 1944; Wilkins, 1989; Green, 1994; Henderson, 1998]. There are also a dictionary [Henderson and Dobson, 1994] and a picture dictionary [Broad, 2008] available.

3.3.2 Nahuatl

Nahuatl belongs to the Aztecan branch of the Uto-Aztecan language family. It is one of the most widely spoken indigenous languages of the Americas, with about 1.5 million speakers, but less than 15% of Nahuatl speakers are monolingual, and Spanish literacy greatly exceeds Nahuatl literacy [Hill and Hill, 1986]. Indigenous populations have become increasingly marginalized in Mexican society; Rolstad [2001] argues that Nahuatl is at serious risk of replacement by Spanish. The Mexican government recognizes 30 distinct varieties of Nahuatl; some varieties have already disappeared and many are severely endangered. The modern varieties show considerable differences, and not all are mutually intelligible. Some small dialects of Nahuatl still remain virtually undocumented [Grinevald, 2008].

Nahuatl describes spatial relations with relational nouns, which always occur after a noun or a possessive prefix. Most modern dialects of Nahuatl have also incorporated Spanish prepositions that compete with the relational nouns. Other examples of Spanish influence on the language include simplification of the morphology and a tendency toward fixed word order. Some varieties of Nahuatl inflect the verb in a way similar to Arrernte associated motion, to show the direction of the verbal action. For example, the prefix *on-* is used for verbs indicating direction of action away from the speaker [Karttunen, 1992]. The use of these inflections has not been studied much, and it warrants further research.

As far as resources, there are several detailed grammars of Classical Nahuatl published by missionaries, including Horacio Carochi's *Arte de la lengua Mexicana*, and there are several modern textbooks and dictionaries of the larger dialects [Endangered Language Alliance]. There are also some electronic resources, including: an online trilingual dictionary between English, Spanish, and Nahuatl [Wood], a preliminary computational model of Nahuatl morphology in Grammatical Framework (GF) [Ashton, 2013], and a formal morphology in XFST [Maxwell and Amith, 2005]. Amith is also developing the Nahuatl Learning Environment, which links lexicon, grammar, and corpus into a multimedia system for research and learning [Amith].

3.4 Overview of WELT

A visual representation of the intended workflow for using WELT is provided in Figure 3.1. At this time, we have used WELT English and WELT L2 only independently of each other, and the workflow as a whole has not been tested in practice. The WELT English tools for scene creation and elicitation are currently useable; the only exception is that adding custom vignettes and custom 2D/3D content requires the assistance of WordsEye developers to make the necessary modifications to WordsEye and VigNet. We have created a prototype of the WELT L2 tools for modeling and documentation and tested it with a small grammar we have developed for Arrernte. This prototype demonstrates how WELT L2 can potentially be used in fieldwork, but the documentation tools will require further development before they can be used more generally. In addition, both WELT English and WELT L2 still rely on the older desktop version of WordsEye. This makes it difficult to release the program to the public for general use, both due to the need to license the 3D objects stored locally on a user's computer and because the desktop application is no longer being supported or updated by the WordsEye developers. In Chapter 5, we will describe some of our work toward updating WELT to use the newest version of WordsEye.

The first step in the workflow is using WELT English to prepare a set of 3D scenes to be used to elicit targeted descriptions or narratives. An important part of this phase is the cultural adaptation of the graphical semantics used in WordsEye, so that scenes will be relevant to the native speakers a linguist works with. We will discuss cultural adaptation in more detail in Section 3.6.1. Next, the linguist works with an informant to generate language data based on prepared 3D scenes. This can be a dynamic process; as new questions come up, a linguist can easily modify existing scenes or create new ones. WELT English also automatically syncs recorded audio with open scenes and provides an interface for the linguist to write notes, textual descriptions, and glosses. After the elicitation session, the linguist can use WELT English to review the data collected, listen to the audio recorded

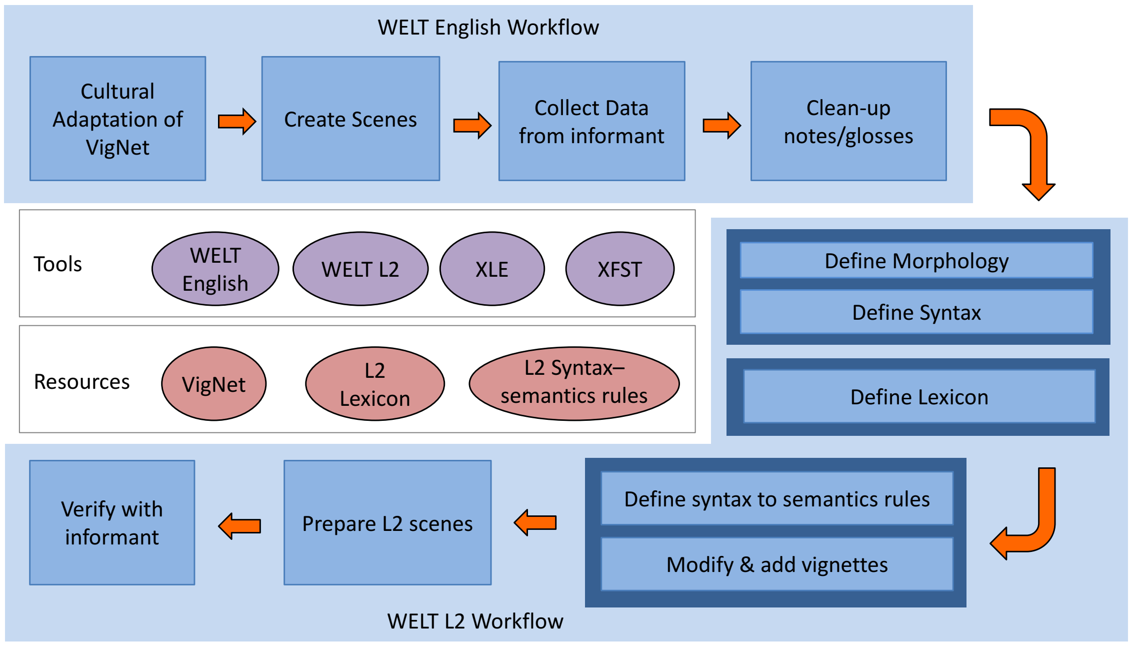


Figure 3.1: WELT workflow

for each scene, and revise notes and glosses. The linguist can then create additional scenes to elicit more data or begin the formal documentation of the language. We will discuss creating scenes and eliciting data with WELT in Section 3.6.2, including examples from our elicitation sessions for Nahuatl.

Creating a text-to-scene system with WELT L2 requires formal models of the morphology, syntax, and semantics of a language. The focus of our work on WELT is on modeling the interface between syntax, lexical semantics, and graphical semantics. Therefore, although WELT requires models of morphology and syntax to generate a text-to-scene system, we are relying on third-party tools to build those models. For our pilot work using WELT L2 to model Arrernte, we have used XFST [Karttunen *et al.*, 1997; Beesley and Karttunen, 2003] to model the morphology and XLE [Crouch *et al.*, 2011] to model the syntax in the LFG (lexical-functional grammar) formalism [Kaplan and Bresnan, 1995]. These are mature systems that we believe are sufficient for the formal documentation of morphology and syntax. We are also researching other options that would be more accessible to non-computational linguists. In Chapter 4, for example, we discuss the possibility of automatically creating a syntactic parser for WELT using annotated dependency structures. It is important to note, though, that the modeling done in WELT L2 does not require a perfect syntactic parser, so the syntactic grammars provided as models do not need to be complex (and WELT L2 provides an interface for selecting among ambiguous syntactic structures as part of its text-to-scene pipeline.) We will discuss our grammar development for Arrernte in Section 3.5. WELT L2 also requires the creation of a lexicon for the endangered language that maps lexical items into VigNet concepts; it provides tools for searching the VigNet ontology to find relevant concepts for the lexicon; we will discuss the lexicon in more detail in Section 3.7.1. WELT L2 also provides user interfaces for modeling the syntax-semantics interface, lexical semantics, and graphical semantics of a language through the creation of syntax-to-semantics rules. We will discuss these in more detail in Section 3.7.2.

Once models of morphology, syntax, and semantics are in place (note that these can be working models, and need not be complete), WELT L2 puts the components together into a text-to-scene system that takes input in the endangered language and uses the formal models to generate pictures. This system can then be used to verify theories with informants and revise grammars if and when an informant indicates that the scene generated for a particular input sentence is not correct. As new questions arise, WELT English can also continue to be used to create elicitation materials and collect linguistic data

3.4.1 Modifications to WordsEye

The WELT tools described in this chapter use the WordsEye desktop application for Mac OS X. The WordsEye desktop application includes a user interface where the user can type simple sentences that are processed to produce a 3D scene. The user can then modify the text to refine the scene. In order to use WordsEye with WELT, some modifications were required, both to the user interface and to the underlying system. The original WordsEye architecture was described in the previous chapter and shown visually in Figure 2.1. The modified WordsEye architecture used by WELT is shown in Figure 3.2.

To adapt WordsEye for use with WELT English, we first made some changes to the user

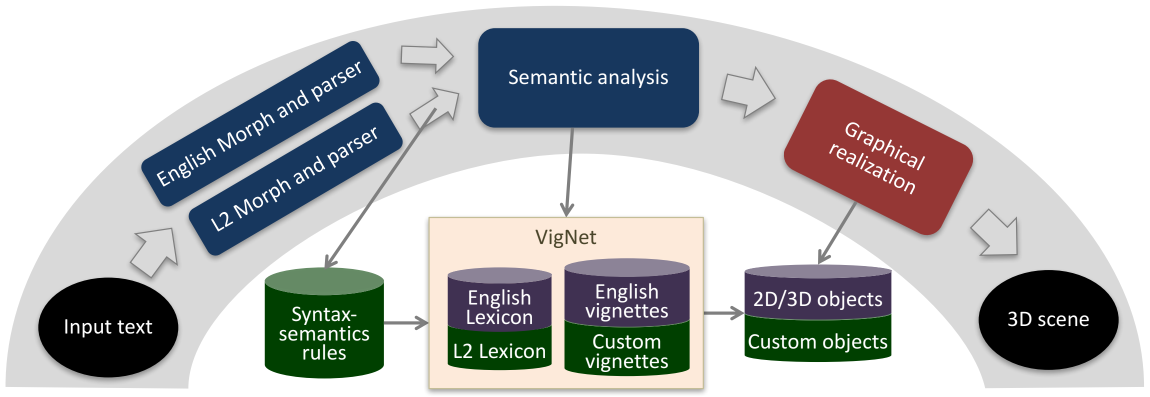
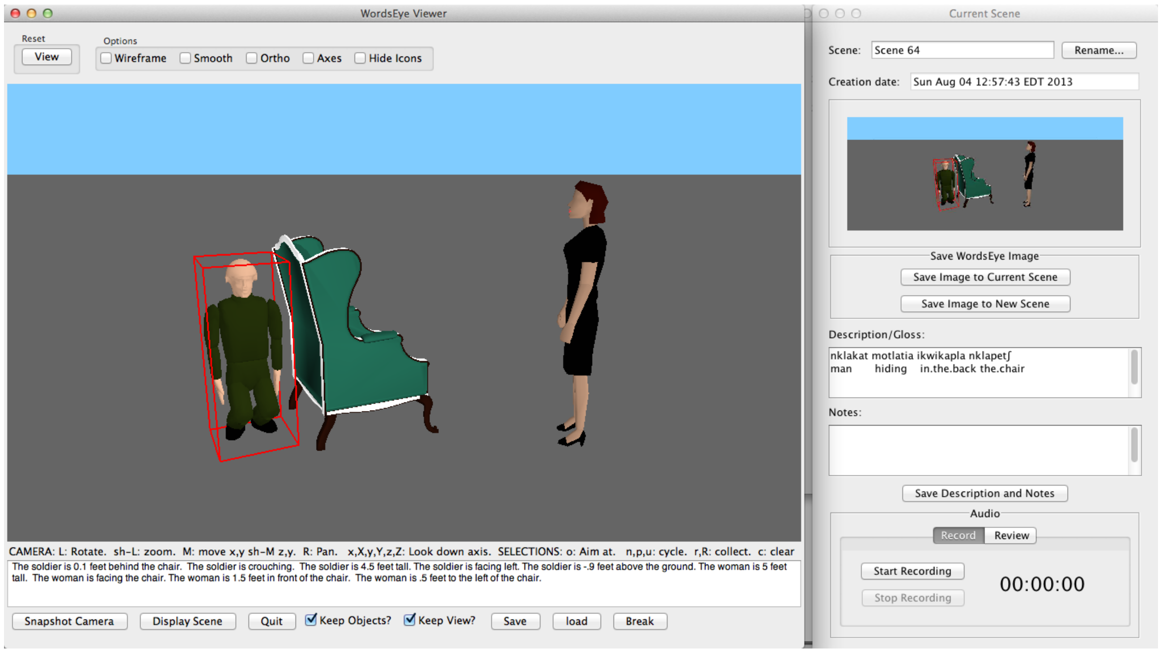


Figure 3.2: WELT architecture

interface of the WordsEye desktop application. To allow WELT users to focus the attention of an informant on particular items in a scene, we modified WordsEye to let individual objects and their parts to be selected and highlighted with a bounding box. We also added the functionality to allow the scene currently open in WordsEye to be accessed by an external application, so that scenes can be saved and reloaded using the WELT user interface. To accomplish this, when WordsEye processes input text in order to generate a scene, it also saves information about the scene to disk, including a JPEG image of the scene, the text used to generate the scene, the position of the camera, and any selected objects that should be indicated with bounding boxes. A screenshot showing the WordsEye interface to the WELT elicitation tools is shown in Figure 3.3. For WELT English, the overall processing pipeline for WordsEye is largely unchanged, taking English text as input and producing a 3D scene representing its meaning. However, in order to allow the cultural adaptation required to create scenes that are maximally relevant to endangered language speakers, we modify VigNet to allow for the addition of custom vignettes and 2D/3D content. For example, for our work on Arrernte, we customize the *kick* vignette to show Australian football rather than soccer, and change the ground texture from grass to desert dirt.

This new culturally relevant content can also be used in the WELT L2 documentation tools. WELT L2 requires further modifications of WordsEye in order to support the creation of a text-to-scene system for the endangered language. When a sentence is processed by WordsEye, it goes through three main stages: (1) morphological analysis and syntactic



CAMERA: L: Rotate. sh-L: zoom. M: move x,y sh-M z,y. R: Pan. x,X,y,Y,z,Z: Look down axis. SELECTIONS: o: Aim at. n,p,u: cycle. r,R: collect. c: clear

The soldier is 0.1 feet behind the chair. The soldier is crouching. The soldier is 4.5 feet tall. The soldier is facing left. The soldier is -.9 feet above the ground. The woman is 5 feet tall. The woman is facing the chair. The woman is 1.5 feet in front of the chair. The woman is .5 feet to the left of the chair.

[✓ Keep Objects?] [✓ Keep View?]

Scene: Scene 64

Creation date: Sun Aug 04 12:57:43 EDT 2013

Description/Gloss:

```
nklakat motlatia ikwikapla nklapetʃ
man     hiding   in.the.back the.chair
```

Notes:

Figure 3.3: WordsEye interface to the WELT elicitation tools. The "Save Image to Current Scene" and "Save Image to New Scene" buttons are used to transfer information about the scene currently open in WordsEye to WELT.

parsing, (2) semantic analysis, and (3) graphical realization. To produce a text-to-scene system for a new language, WELT must replace the English linguistic processing modules with models for the new language. Since our work on Arrernte uses XFST for morphology and XLE for syntax, our modifications to WordsEye for the WELT L2 prototype are based on interfacing with these tools. However, it would be fairly simple to adapt our system to substitute other tools in in the future. When text is input into the WELT L2 text-to-scene system, rather than being passed directly into the normal WordsEye pipeline, it is instead passed to an external script that processes it with XLE and the Arrernte grammar. We have added one additional feature to the morphology and syntax module of the WELT L2 text-to-scene system: an interface that allows the user to select an f-structure from multiple options produced by XLE, in case the grammar is ambiguous. This way, it is still possible use the WELT text-to-scene system to verify semantic documentation even if the syntactic documentation is not complete. We will see an example of this in Section 3.7.3. Once this is done, the f-structure is processed with the syntax-to-semantics rules to produce a lexical-semantic representation that is compatible with WordsEye. This step also requires a lexicon that maps endangered language words into VigNet concepts. The lexical semantic representation is then passed back to WordsEye so that a 3D scene can be generated.

3.5 Grammar Development for Arrernte

We collaborated with researchers at Macquarie University to document Arrernte syntax using LFG [Kaplan and Bresnan, 1995], as part of a project to automatically generate Arrernte text related to Australian football [Lareau *et al.*, 2011; Lareau, 2012]. In LFG, linguistic structure is represented by a parallel, linked combination of a surface-oriented constituent structure (c-structure) and a functional structure (f-structure). The f-structure is a dependency structure that models predicate-argument structure, and a suitable interface to VigNet.[3] The grammar for Arrernte is in two parts, a finite state transducer for the morphology, developed with XFST [Karttunen *et al.*, 1997], and the syntactic grammar developed in XLE [Crouch *et al.*, 2011]. It covers basic sentences and NP structure and a few unusual features of Arrernte: split case pronouns, verbless sentences, associated motion, spatial relationships, and same-subject inflection on the verb [Dras *et al.*, 2012].

In deciding to use LFG as the formalism for the grammar, several factors were considered, including linguistic suitability and availability of resources. One of the main attributes of Arrernte is that it is a non-configurational language (it has a flat phrase structure, allowing syntactically discontinuous expressions, and a relatively free word order), and there has already been a substantial amount of work on using LFG to model non-configurational Australian languages [Simpson, 2007; Nordlinger, 1997; Nordlinger and Bresnan, 2011], making LFG desirable for the Arrernte project in particular. Another point in favor of LFG is that there are already mature and widely used tools for developing grammars. Most

[3] Although it is common with LFG to represent semantics using linear logic and glue semantics [Dalrymple *et al.*, 1993; Dalrymple, 2001], and in fact the Arrernte grammar we use for syntax does have a semantic component that uses glue semantics [Lareau *et al.*, 2012], we do not use that representation. Nor do we need to use LFG's a-structure, since we use an alternative representation for the lexical semantics.

prominent is the Xerox Linguistic Tool (XLE) [XLE Project]. XLE is also the basis for the Parallel Grammar Project (ParGram) [Butt *et al.*, 2002; ParGram / ParSem, 2013], a collaborative effort to develop wide coverage grammars in LFG for multiple languages. These existing grammars and the common standards and tutorials that ParGram provides will also simplify the creation of formal syntactic grammars for WELT L2 as we work toward expanding it for more languages.

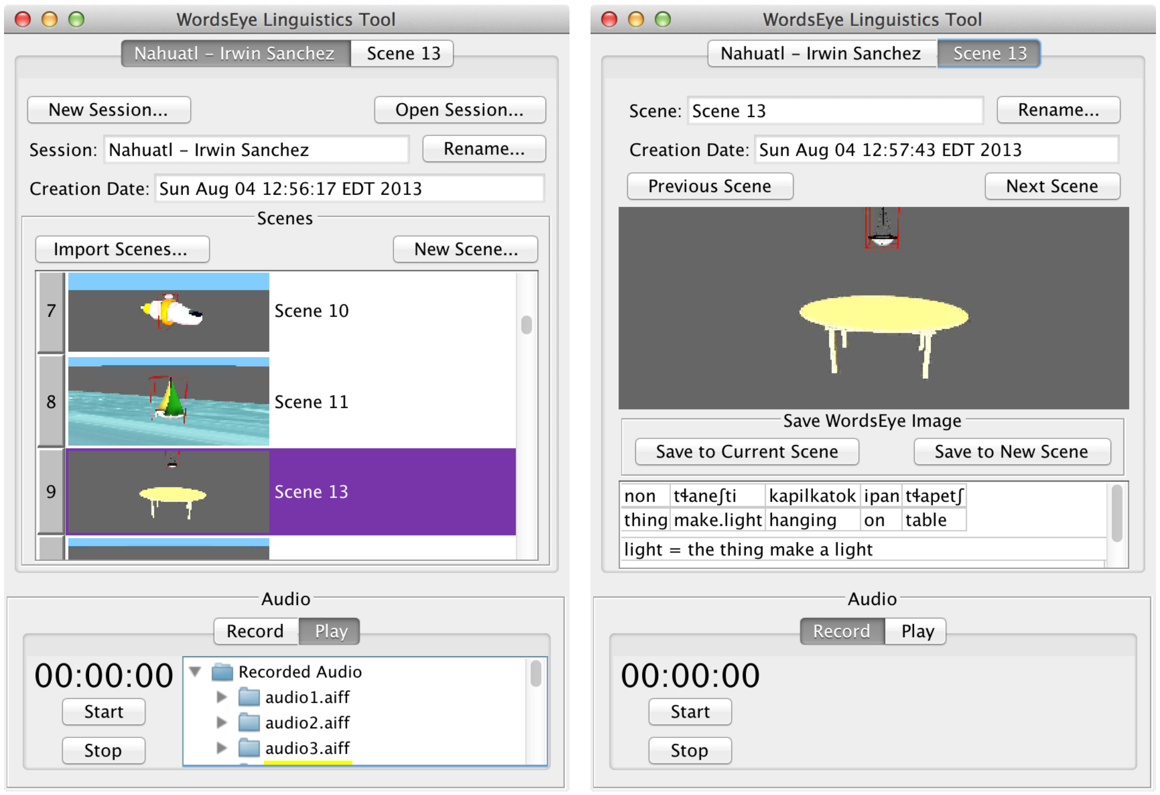
3.6 Elicitation Tools (WELT English)

Each elicitation session in WELT is organized around a set of 3D scenes, which are created by inputting English text into WordsEye. The scene that is currently open in the WordsEye application can be saved and added to the WELT session, as indicated in Figure 3.3. Scenes that have been previously added to the session can be re-opened in WordsEye and modified, either overwriting the original scene or saving the changes as a new scene. Each scene can be annotated with with textual descriptions, glosses, and notes. Audio for the session can be recorded, and the recording is automatically saved and synced with timestamps for the scenes open in WELT. The audio can be played back to review any given scene. Scenes can be imported and exported between sessions, so that useful scenes can be reused and data compared. Screenshots of the WELT main elicitation interface and the interface for annotating a scene are included in Figure 3.4.

3.6.1 Cultural Adaptation of VigNet

As we discussed in Chapter 2, semantics in WordsEye are represented with VigNet, a resource which groups similar lexical items, along with graphical relations, into frames called vignettes. We use VigNet to represent the semantics in WELT.

Large parts of VigNet are language- and culture-independent. The low-level graphical relations used to express graphical semantics are based on physics and human anatomy and do not depend on language. However, the graphical semantics for a vignette may be culture-specific, and some new vignettes may need to be added for a culture. In the United States, for example, the sentence *The woman boiled the water* might invoke a scene with a



Session: Nahuatl – Irwin Sanchez

Creation Date: Sun Aug 04 12:56:17 EDT 2013

7		Scene 10
8		Scene 11
9		Scene 13

00:00:00

Scene: Scene 13

Creation Date: Sun Aug 04 12:57:43 EDT 2013

non	tɬaneʃti	kapilkatok	ipan	tɬapetʃ
thing	make.light	hanging	on	table

light = the thing make a light

00:00:00

Figure 3.4: Screenshots of WELT elicitation interfaces. Left: WELT interface for managing a session. Right: WELT interface for annotating a scene.

pot of water on a stove in a kitchen. Among the Arrernte people, it would instead invoke a woman sitting on the ground in front of a kettle on a campfire. Figure 3.5 shows an illustration from the Eastern and Central Arrernte Picture Dictionary [Broad, 2008] of the sentence *Ipmenhe-ipmenhele kwatye urinpe-ilemele iteme*, "My grandmother is boiling the water." The lexical semantics for the English verb *boil* and the Arrente verb *urinpe-ileme* are the same, the relation APPLY-HEAT.BOIL. However, the vignettes map to different, culture-typical graphical semantics. Figure 3.6 shows the instantiated vignettes for our example, demonstrating the cultural differences in the graphical semantics. To handle cultural differences like these, VigNet needs to be extended with new graphical semantics for existing vignettes that need to be modified, and new vignettes for scenarios not already covered. Currently, these modifications to VigNet must be done by the WordsEye developers. Future

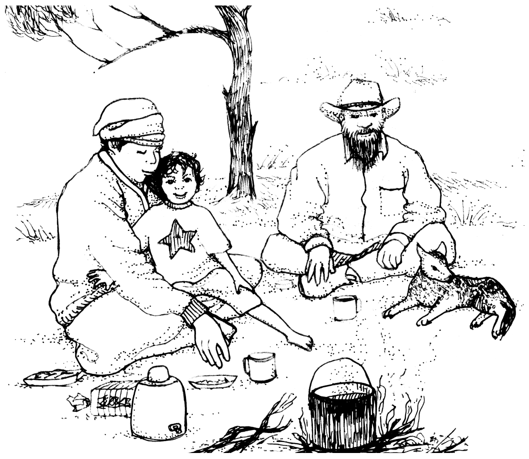


Figure 3.5: Illustration of Arrernte sentence *Ipmenhe-ipmenhele kwatye urinpe-ilemele iteme*, “My grandmother is boiling the water.” [Broad, 2008]

work on WELT would allow users to make these changes themselves.

3.6.1.1 Custom WordsEye Objects

Another way to adapt WordsEye to a culture or region is to add relevant 3D objects to the database. WordsEye also supports 2D-cutout images, which is an easy way to add new material without 3D modeling. In the WordsEye desktop application, each new object has to be manually incorporated into WordsEye by the WordsEye developers. In the newer web application version of WordsEye, users can upload custom content directly. In Chapter 5, we will discuss the newer version of WordsEye in more detail, including its application to the WELT elicitation tools.

We have created a corpus of 2D and 3D models for WordsEye that are specifically relevant to aboriginal speakers of Arrernte, including native Australian plants and animals and culturally relevant objects and gestures. Many of the pictures we created are based on images from IAD Press, used with permission. Most of the original IAD Press images were black-and-white drawings similar to the one in Figure 3.5. We enhanced and cropped these in PhotoShop. Examples of our enhanced images are shown in Table 3.1; the complete collection of enhanced images is provided in Appendix A. Some scenes that use these images are included in Figure 3.7.

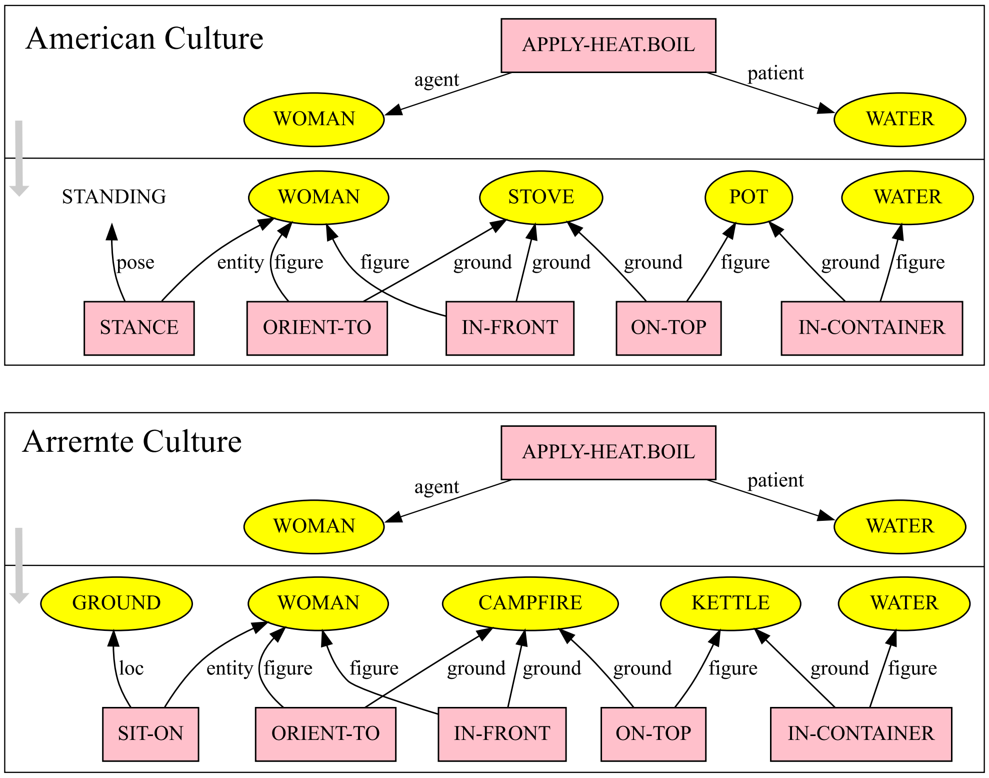


Figure 3.6: (Instantiated) vignettes for *the woman boils the water.* The high-level semantics of APPLY-HEAT.BOIL are decomposed into sets of objects and primitive graphical relations that depend on cultural context.

3.6.2 Preparing Scenes and Eliciting Data: Nahuatl Topological Relations

To evaluate WordsEye's usefulness in the creation of pictures for eliciting spatial language, we created a set of scenes based on the Max Planck topological relations picture series [Bowerman and Pederson, 1992]. Some examples of pictures from the Max Planck topological relation series are shown in Table 3.2. Many of the scenes were created in WELT by two undergraduate students who were pursuing majors in linguistics. These students did not have any background or experience in computational linguistics or computer science, thus demonstrating WELT's usability by non-computational linguists. In creating the scenes, we used the new feature of WordsEye described in Section 3.4.1, that allows users to highlight

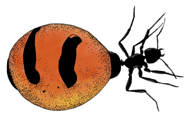
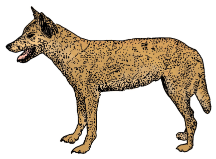
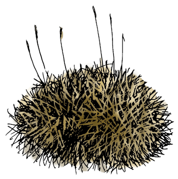
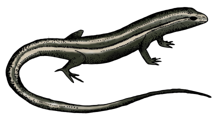
			
(a) Honey Ant (*yerrampe*)	(b) Dingo (*akngwelye artnwere*)	(c) Spinifex (*aywerte*)	(d) Skink (*ikwarre*)

Table 3.1: Images created for Arrernte WELT (Arrernte translations in parentheses)

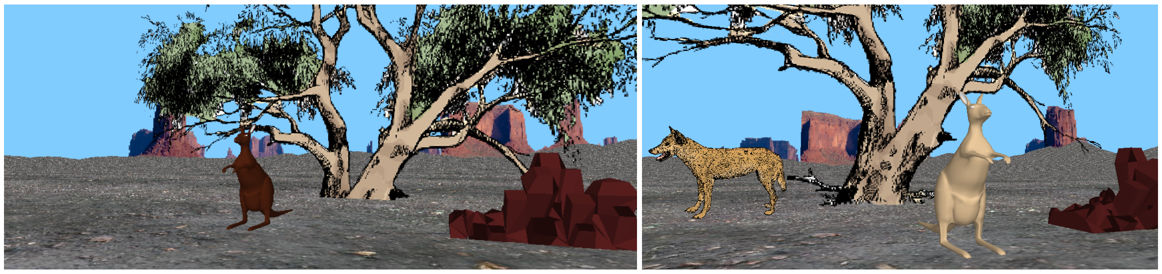
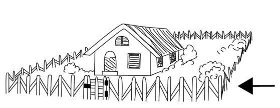
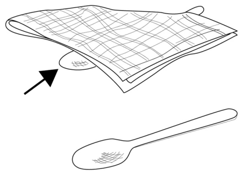


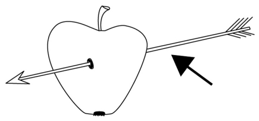
Figure 3.7: WordsEye scenes using custom 2D gum tree and dingo from our image corpus

specific objects (or parts of objects) in a scene. We were able to recreate 40 out of the 71 pictures in the series. One of the main issues that prevented us from creating the full set was that WordsEye does not currently have the objects needed to produce the desired scene. There were also cases where the graphical functionality of WordsEye needs to be enhanced to allow more precise positioning of objects. We used these scenes to elicit descriptions from a native speaker of Nahuatl; some examples of scenes and descriptions are included in Table 3.3. Section B.1 shows the full set of scenes we created along with the input text used to generate the scenes in WordsEye. For scenes that were not reproduced in WordsEye, we also briefly describe the reasons we were unable to do so.[4] Section B.2 shows the Nahuatl descriptions we elicited for each scene.

[4] Note that many of these problems, particularly those involving missing 3D objects, will be fixed when WELT has been updated to use the latest version of WordsEye. For example, WordsEye now has a number of 3D objects for bugs, which we were missing in pictures (7) and (52), as well as a 3D object of an open bag with handles, which was needed for pictures (14) and (66). Other issues will be addressed as WordsEye developers continue to add new 3D objects and implement more graphical functionality.


(a)


(b)


(c)

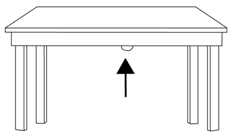
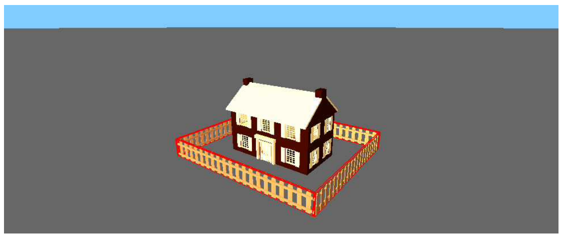
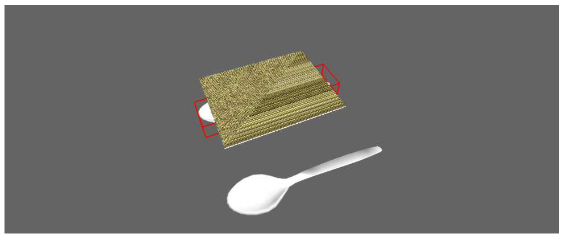

(d)

Table 3.2: Pictures from the Max Planck Topological Relations Picture Series [Bowerman and Pederson, 1992]



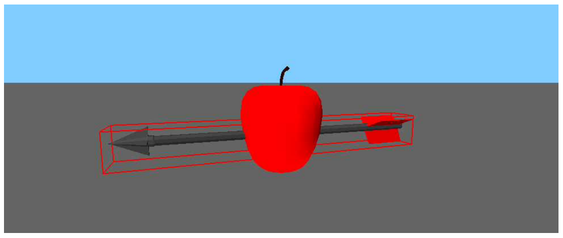
(a)

in	*tapametł*	*tłatsakwa*	*se*	*kali*
the	fence	around	the	house



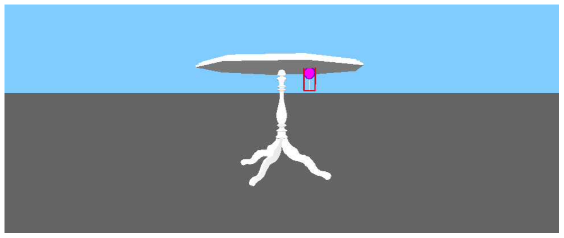
(b)

in	*amatł*	*tłakentija*	*se*	*kutʃara*
the	paper	cover	one	spoon



(c)

in	*kwawitł*	*tłapanawi*	*tłakoja*	*se*	*mansana*
the	stick	pass.thru	in.middle	one	apple



(d)

in	*tsopelik*	*katsekotok*	*tłatsintła*	*in*	*tłapitʃ*
the	candy	sticking	under	the	table

Table 3.3: Nahuatl elicitations

3.7 Documentation Tools (WELT L2)

WELT L2 provides the tools to formally document the semantics of a language. It also uses this documentation to automatically generate a text-to-scene system for the language. The formal documentation allows precise description of the lexical semantics of a language.

Because WELT is centered around the idea of 3D scenes, the formal documentation will tend to focus on the parts of the semantics that can be represented graphically. Note that this can include figurative concepts as well, although the visual representation of these may be culture-specific. However, users do not necessarily need to be limited by the graphical output; WELT can be used to document other aspects of semantics as well, but it will not be possible to verify these theories using the text-to-scene system. In this section, we will describe the user interface for documenting semantics, as well as a text-to-scene system for Arrernte created with WELT.

In order to create a text-to-scene system for an endangered language, WELT requires the components shown in Figure 3.2. The custom vignettes and 3D objects will largely have been done during the cultural adaptation of VigNet described in Section 3.6.1. In addition to these, the system requires a morphological analyzer, syntactic parser, a lexicon that maps endangered language words into vignette concepts, and a set of syntax-to-semantics rules which map the output of the syntactic parser into vignettes. We discussed our development of morphological and syntactic grammars for Arrernte in Section 3.5. In this section, we will discuss creating the lexicon and the syntax-to-semantics rules.

3.7.1 The Lexicon and Ontology Browser

The lexicon in WELT is a list of word forms mapped to semantic concepts, which allows nouns from the endangered language to be converted into graphical objects. WELT includes a visual interface for searching VigNet's ontology for semantic concepts and browsing through the hierarchy to select a particular category. The ontology browser is used in several parts of WELT, including the tools for creating the lexicon and and the tools for modifying the constraints in syntax to semantics rules. We have created a mapping for the lexical items in the Arrernte grammar; a partial mapping is shown in Table 3.4. As an example, to find an appropriate concept for the Arrernte word *panikane* 'cup', we can search the ontology browser for *cup*. Figure 3.8(a) shows the portion of the ontology that results from this search. Semantic categories are displayed one level at a time, so initially only the concepts directly above and below the search term are shown. Selecting another visible

Lexical Item	**VigNet Concept**
artwe	PERSON.N
panikane	CUP.N
angepe	CROW.N
akngwelye	DOG.N
apwerte	ROCK-ITEM.N
tipwele	TABLE.N

Table 3.4: A mapping from nouns (lexical items) to VigNet semantic concepts

node in the graph results in its own immediate neighbors being displayed. Figure 3.8(b) shows the result of selecting DRINKING-CUP.N. Here, we have decided to map *panikane* to CUP.N.

3.7.2 Documenting Semantics

The goal of WELT L2 is to provide the means to formally document the semantics of a language and create a text-to-scene system for that language. The formal documentation allows precise description of the lexical semantics of a language. The WELT L2 semantics is represented using VigNet, which has been developed for WordsEye based on English. To use the WordsEye architecture, the system needs to be able to map between the syntax of the endangered language and a representation of semantics compatible with VigNet. Most obviously, the lexical items and valence patterns are different for other languages. One instance showing why this is necessary occurs in our example Arrernte sentence. When discussing football in English, one would say that someone *kicks a goal* or *makes a goal.* In Arrernte, one would say *goal arrerneme*, which is a light verb construction that translates literally to "put a goal." Although the semantics of both sentences are the same, the entry for *put* in the English VigNet does not include this meaning, but the Arrernte text-to-scene system needs to account for it. Another issue is the motion-related inflections that can be applied to verbs in both Arrernte and Nahuatl; the valence patterns used by the WELT L2

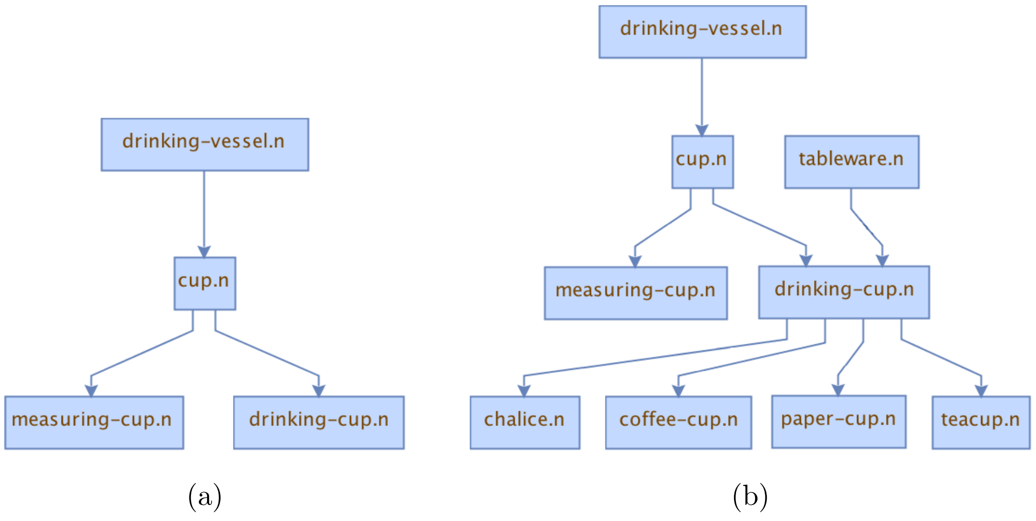


Figure 3.8: Screenshots of the ontology browser

text-to-scene system need to handle these inflectional variations, which are not present in English.

To address such instances, the linguist uses WELT L2 to define lexical, syntactic, and semantic information. This begins with the creation of a mapping from the lexicon into VigNet concepts, so that the noun strings in the endangered language can be converted to graphical objects. We discussed creating the lexicon in Section 3.7.1. To handle valence patterns, WELT includes an interface for the linguist to specify a set of rules that map from syntax to (lexical) semantics. The interface allows users to develop rules that map the lexical structure of the new language into a high-level semantic representation compatible with VigNet. Since we are modeling Arrernte syntax with LFG, the rules currently take syntactic f-structures as input, but the system could easily be modified to accommodate other formalisms. Rules are specified by defining a tree structure for the left-hand (syntax) side and a directed acyclic graph for the right-hand (semantics) side. The left-hand side of a rule consists of a set of conditions on the f-structure elements and the right-hand side is the desired semantic structure.

As an example, we construct a rule to process sentence (5):

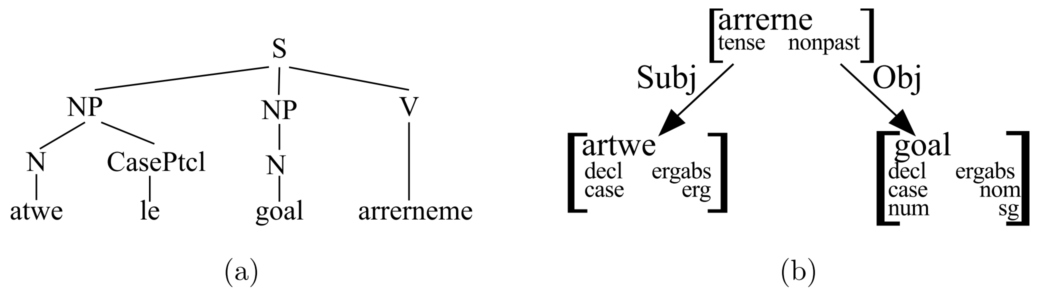


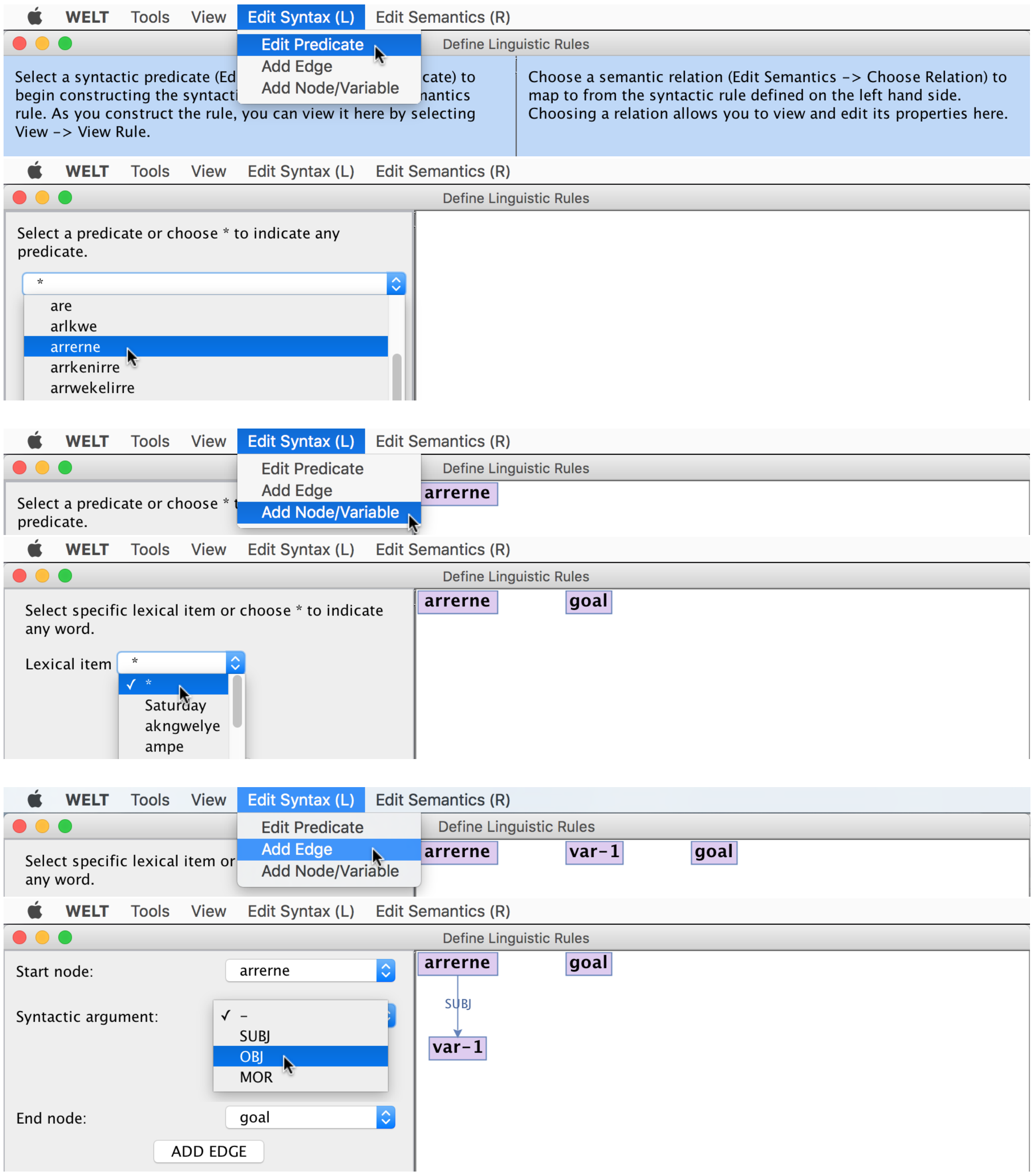
Figure 3.9: (a) c-structure and (b) f-structure for *artwe le goal arrerneme.*

(5)	*artwe*	*le*	*goal*	*arrerneme*
	man	ERG	goal	put.NPP

‘The man kicks a goal.’

We begin by creating the left-hand (syntax) side of the rule, by specifying a tree structure for the syntactic constraints. Screenshots demonstrating this process in WELT are shown in Figure 3.10. For this sentence, our Arrernte grammar produces the f-structure in Figure 3.9(b). We could create a rule specifying the specific lexical item *artwe* in the subject position; to generalize somewhat, we create a rule that selects for predicate *arrerne* with object *goal* and any subject. First, we specify the syntactic predicate, choosing *arrerne* from the dropdown menu. Next, we add nodes for each of the syntactic arguments that will be defined in our rule. These can be specific lexical items or they can be variables which will be replaced with lexical items when an actual sentence is processed. We then connect these nodes with edges by specifying the start and end nodes and the syntactic grammatical relation between them. The values in the dropdown menus in Figure 3.10 are extracted from the XLE source files for the Arrernte (LFG) grammar.

After we have finished specifying the syntax, we create the right-hand (semantics) side of the rule. Screenshots demonstrating creating the right-hand side of our rule in WELT are shown in Figure 3.11. We begin by searching VigNet for a relevant vignette. In this case, we search for *kick* and select the vignette `we.kick-goal.vg`. WELT then displays the default semantic constraints for the arguments vignette, which we can edit. We first modify the `agent` argument of the vignette, to specify that `var-1` should be mapped to this semantic



Select a syntactic predicate (Ed cate) to begin constructing the syntacti nantics rule. As you construct the rule, you can view it here by selecting View –> View Rule.

Choose a semantic relation (Edit Semantics –> Choose Relation) to map to from the syntactic rule defined on the left hand side. Choosing a relation allows you to view and edit its properties here.

Select a predicate or choose * to indicate any predicate.

Select specific lexical item or choose * to indicate any word.

Start node: arrerne

Syntactic argument: ✓ –

End node: goal

Figure 3.10: Creating the left-hand (syntax) side of a syntax-to-semantics rule in WELT: selecting a predicate, adding nodes to the syntax tree, and adding edges to specify syntactic arguments.

role. We can also add cultural and geographic customizations at this stage. First, we change the vignette to reflect Australian football. We specify that the target should be FOOTY-GOALPOSTS.N and the projectile should be FOOTY-BALL.N. We also modify the vignette to reflect the geography of the area, changing background-texture from CITY.N to BUTTE.N and global-ground-texture from GRASS-FIELD-SUBSTANCE.N to DESERT-DIRT.N. The values we modified are highlighted in Figure 3.11. Custom semantic categories are selected for the vignette arguments using the same ontology browser we used to create the lexicon.

The completed rule is shown in Figure 3.12. Note that `var-1` on the left-hand side becomes VigNet(`var-1`) on the right-hand side; this indicates that the lexical item found in the subject position of the input should be mapped into a semantic concept using the L2 lexicon.

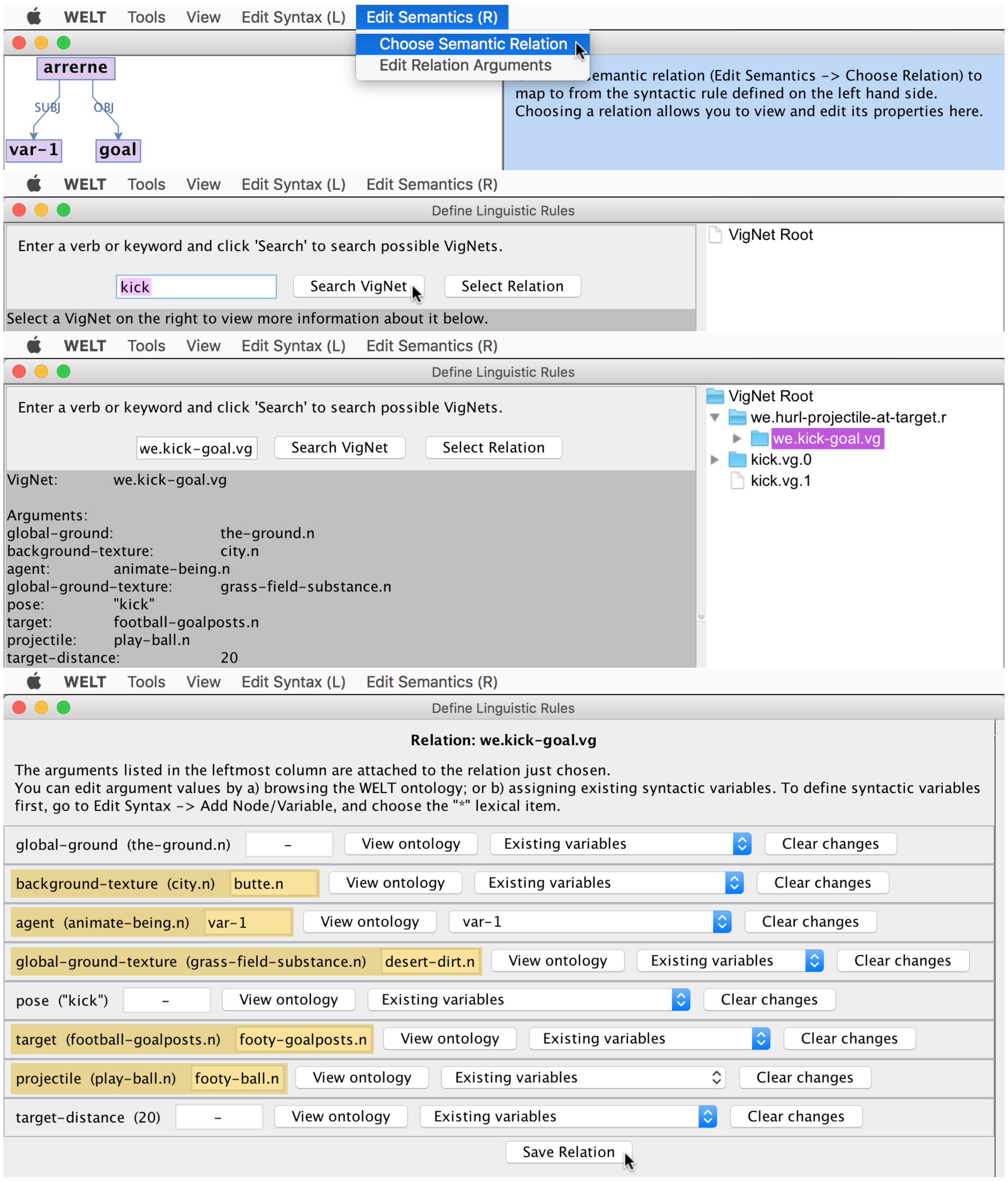
3.7.3 WELT L2 Text-to-Scene Generation

As we discussed in Section 3.4.1, WELT L2 includes a modified WordsEye user interface which can generate a 3D scene from endangered language input. Having created a basic Arrernte lexicon and a simple syntax-to-semantics rule, we have the necessary components to use the modified WordsEye pipeline. We will now walk through the modules of this pipeline, using sentence (5) from the previous section as our input text, reproduced for convenience here:

(5)	*artwe*	*le*	*goal*	*arrerneme*
	man	ERG	goal	put.NPP

‘The man kicks a goal.’

Morphology: The first step of processing the sentence is to run each word through the morphological analyzer. For our Arrernte project, we process the morphology of the example sentence by running each word through the morphological analyzer in XFST. This transforms the verb *arrerneme* into ‘arrerne+NONPAST.’ The other tokens in the sentence remain unchanged.



…emantic relation (Edit Semantics -> Choose Relation) to map to from the syntactic rule defined on the left hand side. Choosing a relation allows you to view and edit its properties here.

Enter a verb or keyword and click 'Search' to search possible VigNets.

Select a VigNet on the right to view more information about it below.

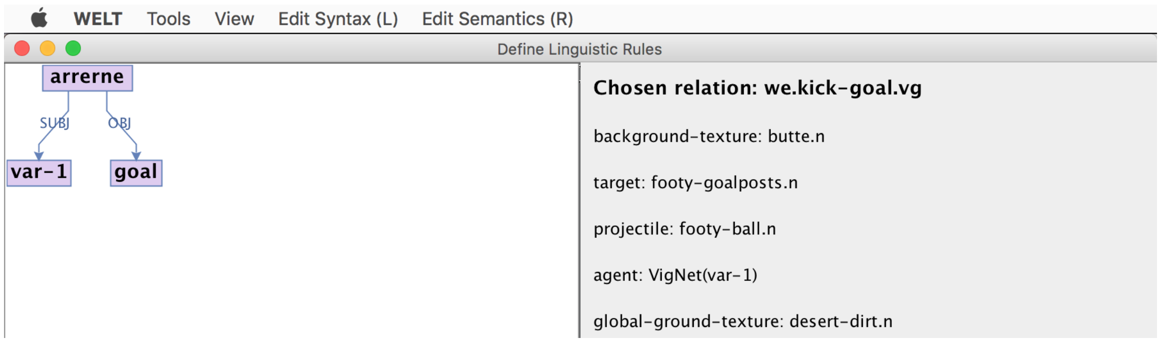
```
VigNet:          we.kick-goal.vg

Arguments:
global-ground:              the-ground.n
background-texture:         city.n
agent:           animate-being.n
global-ground-texture:      grass-field-substance.n
pose:            "kick"
target:          football-goalposts.n
projectile:      play-ball.n
target-distance:            20
```

Relation: we.kick-goal.vg

The arguments listed in the leftmost column are attached to the relation just chosen.
You can edit argument values by a) browsing the WELT ontology; or b) assigning existing syntactic variables. To define syntactic variables first, go to Edit Syntax -> Add Node/Variable, and choose the "*" lexical item.

global-ground (the-ground.n)	-	View ontology	Existing variables	Clear changes
background-texture (city.n)	butte.n	View ontology	Existing variables	Clear changes
agent (animate-being.n)	var-1	View ontology	var-1	Clear changes
global-ground-texture (grass-field-substance.n)	desert-dirt.n	View ontology	Existing variables	Clear changes
pose ("kick")	-	View ontology	Existing variables	Clear changes
target (football-goalposts.n)	footy-goalposts.n	View ontology	Existing variables	Clear changes
projectile (play-ball.n)	footy-ball.n	View ontology	Existing variables	Clear changes
target-distance (20)	-	View ontology	Existing variables	Clear changes

Figure 3.11: Creating the right-hand (semantics) side of a syntax-to-semantics rule in WELT: searching for and selecting a semantic relation (vignette); editing the relation arguments.



Chosen relation: we.kick-goal.vg

background-texture: butte.n

target: footy-goalposts.n

projectile: footy-ball.n

agent: VigNet(var-1)

global-ground-texture: desert-dirt.n

Figure 3.12: A syntax-to-semantics rule created in WELT. Syntactic constraints are represented by a tree-structure on the left side of the rule; semantics are represented by the vignette on the right-hand side.

Syntax: The next step is syntactic parsing via our Arrernte grammar, using XLE. Since our LFG grammar is ambiguous, XLE produces several possible f-structures. WELT L2 displays all the possibilities to the user, so that the correct one can be selected. In XLE, ambiguities in the f-structure are indicated by labeling constraints with tags that indicate which contexts the constraints are defined in. These tags are of the form <a> vs. <∼a>, <b> vs. <∼b>, <c> vs. <∼c>, and so on [Crouch *et al.*, 2011, "Printing Charts"]. WELT displays these tags as prefixes on the node label, and further distinguishes the possible contexts by displaying them in different colors. Figure 3.13 shows what this looks like for our sentence. In this case, the red <∼a> context is the correct one, because *artwe* should be the subject of our sentence. Selecting this context results in the c-structure and f-structure shown in Figure 3.9. The f-structure is passed on to the semantics module.

Semantics: We now walk through the semantic processing of the sentence, assuming a set of rules consisting solely of the one given in Figure 3.12 and the partial noun mapping from Table 3.4 as our lexicon. The f-structure in Figure 3.9(b) has main predicate *arrerne* with two arguments; the object is *goal*. Therefore, it matches the left-hand-side of our rule. The output of the rule specifies the vignette WE.KICK-GOAL.VG. To determine the `agent`, we need to find the VigNet concept corresponding to `var-1`, which occupies the subject position in the f-structure. The subject in our f-structure is *artwe*, and according to Table 3.4, it maps

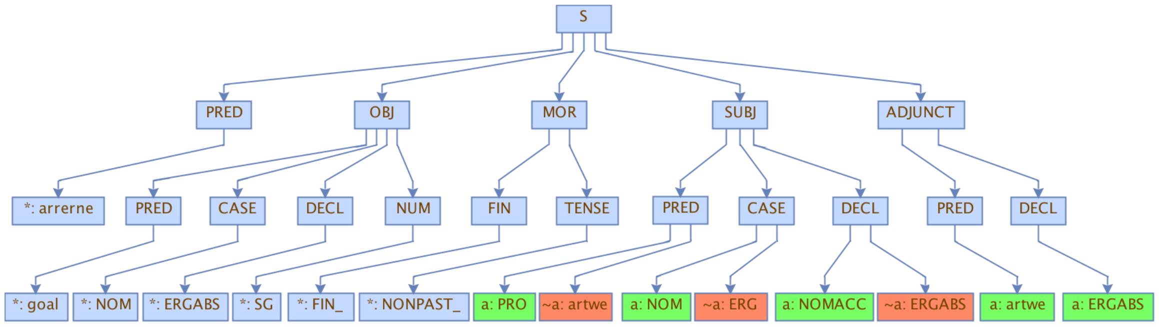
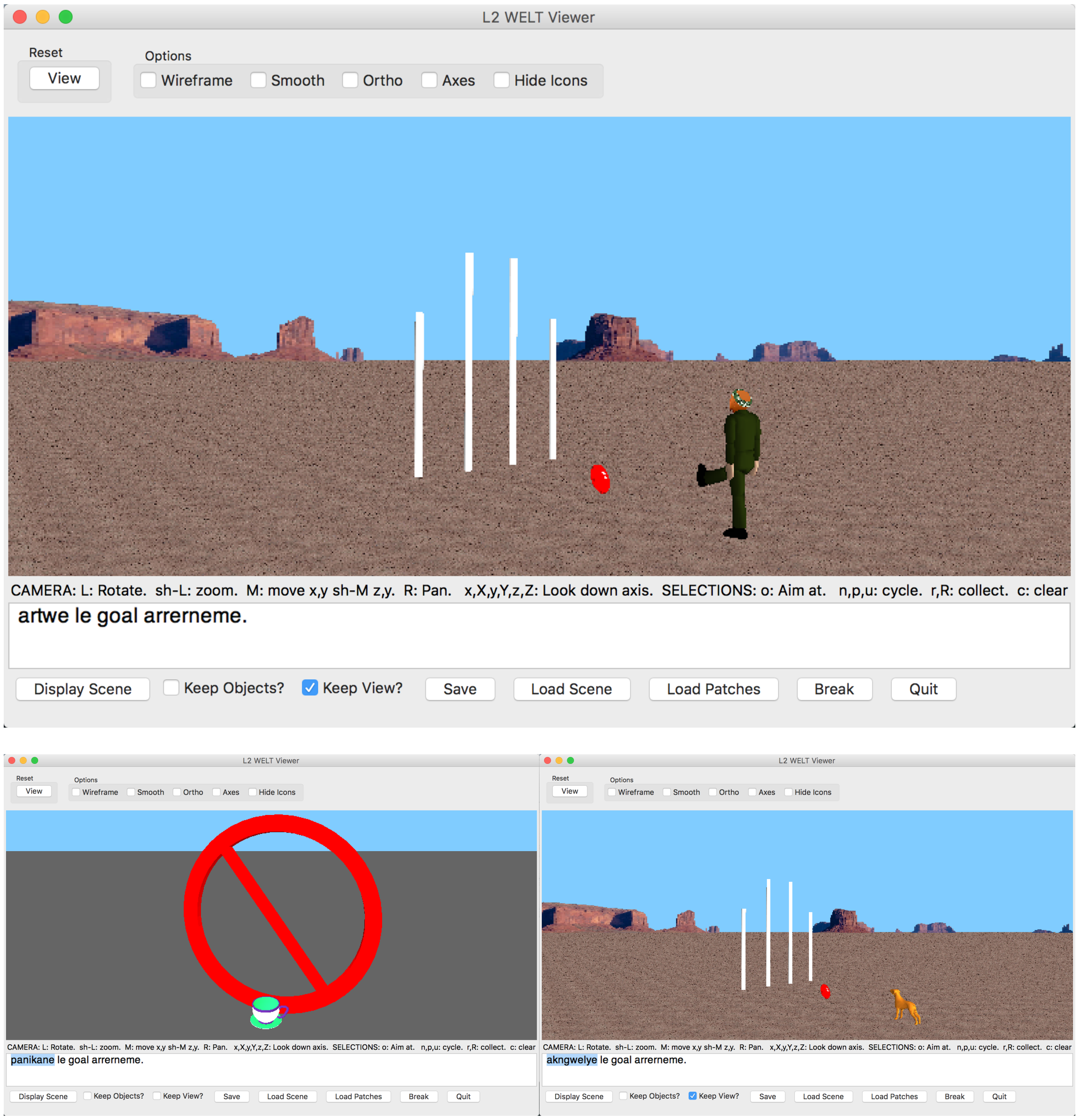


Figure 3.13: Selecting among possible f-structures produced by XLE from an ambiguous grammar.

to the VigNet concept PERSON.N. The resulting semantic representation is augmented with the rest of its graphical semantics, taken from the vignette definition. The WordsEye system then builds the scene from these constraints and renders it in 3D. Screenshots of the WELT L2 WordsEye interface for generating scenes from Arrernte text are shown in Figure 3.14. The top shows the scene generated from our example sentence. The two screenshots on the bottom demonstrate what happens when we substitute other lexical items for *artwe* in the subject position. On the left, *panikane* 'cup' is not an ANIMATE-BEING.N, so the semantic constraints of the vignette are not met. On the right, *akngwelye* 'dog' is an ANIMATE-BEING.N, so WELT L2 is able to generate the scene.

3.8 Future Work: Investigation of Case in Arrernte

One of our goals for using WELT is to study the relationship between the meaning of a sentence and the case of the nouns in it. The relationship in Arrernte between case and semantic interpretation of a sentence is a topic that could be easily explored with WELT. It is possible to significantly alter a sentence's meaning by changing the case on an argument. For example, the sentences in (6) from Wilkins [1989] show that adding dative case to the direct object of the sentence changes the meaning from shooting at and subsequently hitting the kangaroo, to shooting at the kangaroo and *not* hitting it. Wilkins calls this the "dative of attempt."



CAMERA: L: Rotate. sh-L: zoom. M: move x,y sh-M z,y. R: Pan. x,X,y,Y,z,Z: Look down axis. SELECTIONS: o: Aim at. n,p,u: cycle. r,R: collect. c: clear

artwe le goal arrerneme.

[✓ Keep View?]

panikane le goal arrerneme.

akngwelye le goal arrerneme.

[✓ Keep View?]

Figure 3.14: WELT L2 interface for generating a scene from Arrernte text

(6)	a.	*re*	*aherre*	*tyerre-ke*
		3SG	kangaroo	shot-PC

‘He shot the kangaroo.’

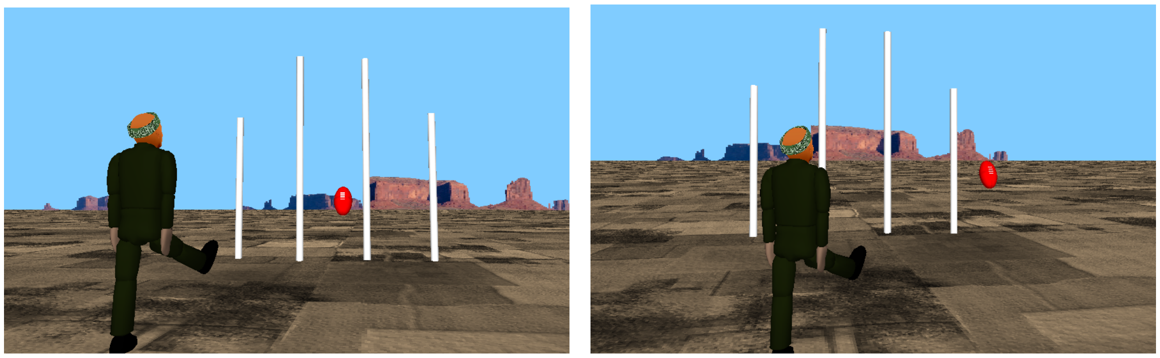


Figure 3.15: WordsEye scenes to elicit the "dative of attempt." In the first scene, the man kicks the ball through the goal; in the second, the man kicks the ball toward the goal but misses.

b.	*re*	*aherre-ke*	*tyerre-ke*
	3SG	kangaroo-DAT	shot-PC

'He shot at the kangaroo (but missed).'

With the help of Myfany Turpin, a linguist who studies Arandic languages, we collected a set of Arrernte sentences, primarily from Broad [2008] and Wilkins [1989], that are interesting in terms of spatial language or case. We created a FieldWorks project for Arrernte that includes all these sentences, translated them, and glossed them at the morphological level. We also entered all of the phonological information necessary for the Fieldworks phonological parser. These sentences can now easily be searched either at the surface level or by the glossed morphemes, so they will be able to be used in future work on Arrernte. We include these glossed and translated sentences in Appendix C.

In order to see how this example generalizes, we can use WELT to create pairs of pictures, one in which the object of the sentence is acted upon, and one in which the object fails to be acted upon. These pictures would be used to elicit descriptions from a native speaker of Arrernte. Figure 3.15 shows a pair of scenes contrasting an Australian football player scoring a goal with a player aiming at the goal but missing the shot. The sentences in (7) are two ways of saying "score a goal" in Arrernte; the scenes in Figure 3.15 would be used to see if a native Arrernte speaker would add the dative of attempt to *goal*, using *goal-ke arrerneme* or *goal-ke kickemileke* when describing the second picture.

(7) a.

artwe	*le*	*goal*	*arrerne-me*
man	ERG	goal	put-NPST

‘The man kicks a goal.’

b.

artwe	*le*	*goal*	*kick-eme-ile-ke*
man	ERG	goal	kick-ENG.TR-CAUS-PST

‘The man kicked a goal.’

3.9 Conclusion

We have described a novel tool designed to assist linguists working with endangered languages. WELT provides useful tools for field linguistics and language documentation, from creating elicitation materials, to eliciting data, to formally documenting a language. It includes a new way to elicit data from informants and an interface for formally documenting the lexical semantics of a language, which in turn allows the creation of a text-to-scene system for a language. In the following chapters, we will revisit some aspects of WELT, including exploring methods of acquiring a syntactic parser without requiring the user to understand particular grammar formalisms (Chapter 4), and extending the WELT English scene creation tools to allow the user to specify more precise configurations by directly modifying the underlying semantics of a scene (Chapter 5).

Chapter 4

Incrementally Learning a Dependency Parser to Support Language Documentation in Field Linguistics

4.1 Introduction

The WordsEye Linguistics Tools (WELT) introduced in Chapter 3 focus on the documentation of semantics. However, in order to make full use of the workflow, including verification of hypotheses via text-to-scene generation, it is also necessary for the system to be able to parse input text into its morphology and syntax. Unfortunately, the ability to easily document syntax is largely missing from existing documentation tools. In this chapter,[1] we perform experiments to test the feasibility of an alternative method of producing a syntactic dependency parser for a language, modeled on the tools for documenting morphology in SIL FieldWorks Language Explorer (FLEx) [SIL International]. FLEx is one of the most widely-used toolkits for field linguists. An important part of FLEx is its "linguist-friendly" morpho-

[1]Some of the work in this chapter was previously published in Ulinski *et al.* [2016a].

logical parser [Black and Simons, 2008], which is fully integrated into lexicon development and interlinear text analysis. The parser is constructed "stealthily," in the background, and can help a linguist by predicting glosses and morphological analyses for interlinear texts. The experiments described in this chapter demonstrate that it will be possible to create a similar tool for syntax in the future.

This chapter makes three contributions. First, we introduce a new corpus of English, Spanish, German, and Egyptian Arabic descriptions of spatial relations and motion events, which we have annotated with dependency structures and other linguistic information. We focused on spatial relations and motion because one of the primary functions of WELT will be to assist field linguists with elicitation of spatial language and documentation of spatial and motion semantics. The corpus is available to the public.[2] [Ulinski *et al.*, 2016b]. Second, we compare the performance of two existing dependency parsing packages, MST-Parser [McDonald *et al.*, 2006] and MaltParser [Nivre *et al.*, 2006], using incrementally increasing amounts of this training data. We find that parsers trained using MaltParser achieve the best performance. Third, we show that using a parser trained on small amounts of data can assist with dependency annotation. We find that even when the parser is trained on a single sentence from the corpus, annotation time significantly decreases.

In Section 4.2, we discuss related work. In Section 4.3, we describe the new publicly available corpus. In Section 4.4, we describe the parsing experiments and discuss the results. Section 4.5 discusses initial experiments with nonlexical models. We discuss the annotation experiments and results in Section 4.6. We conclude and discuss future work in Section 4.7.

4.2 Related Work

There have been a number of investigations into multilingual dependency parsing. For example, Nivre *et al.* [2007b] presents detailed results for 11 languages using the arc-eager deterministic parsing algorithm included in MaltParser. However, results are reported only for the parser trained on the full training set and would not generalize to situations where training data is limited. Likewise, the 2006 and 2007 CoNLL shared tasks of multilingual

[2] `https://doi.org/10.7916/D8W959HJ`

dependency parsing [Buchholz and Marsi, 2006; Nivre *et al.*, 2007a] relied on the existence of ample training data for the languages being investigated. Our work differs in that we are interested in the performance of a dependency parser trained on very little data.

Duong *et al.* [2015] approach dependency parsing for a low-resource language as a domain adaptation task; a treebank in a high-resource language is considered out-of-domain, and a much smaller treebank in a low-resource language is considered in-domain. They jointly train a neural network dependency parser to model the syntax of both the high-resource and the low-resource language. In this paper, we focus on the alternate approach of training directly on small amounts of data.

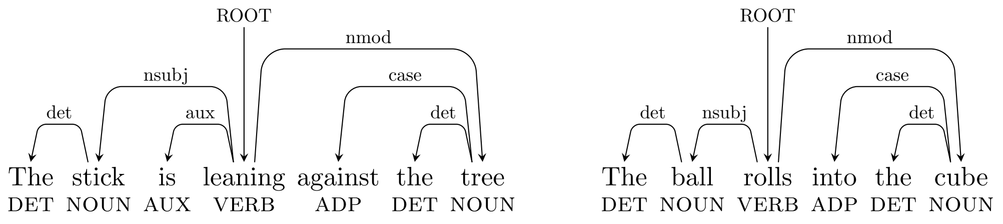
Guo *et al.* [2015] also investigate inducing dependency parsers for low-resource languages using training data from high-resource languages. They focus on lexical features, which are not directly transferable among languages, and propose the use of distributed feature representations instead of discrete lexical features. Lacroix *et al.* [2016] describe a method for transferring dependency parsers across languages by projecting annotations across word alignments and learning from the partially annotated data. However, both of these methods rely on large amounts of (unannotated) data in the target language in order to learn the word embeddings and alignments. It is unclear how well these approaches would work in the context of an endangered language where large amounts of unannotated text will not be available.

Our work also differs from the above because our goal is to incorporate a parser into tools for field linguists studying endangered languages. Currently, there are limited options for creating a syntactic parser for an endangered language. The *ParGram* project [ParGram / ParSem, 2013] aims to produce wide coverage grammars for a variety of languages, but doing so requires knowledge both of the LFG formalism and the XLE development platform [Crouch *et al.*, 2011]. It is unlikely that a field linguist would have the grammar engineering skills necessary to create a grammar in this way. Similarly, the LinGO Grammar Matrix [Bender *et al.*, 2002] is a framework for creating broad-coverage HPSG grammars. The Grammar Matrix facilitates grammar engineering by generating “starter grammars” for a language from the answers to a typological questionnaire. Extending a grammar beyond the prototype, however, does require extensive knowledge of HPSG, making this tool more feasibly used by computational linguists than by field linguists. Our work differs from both ParGram and the Grammar Matrix because we will not require the field linguist to master a particular grammar formalism. Instead, linguists will create a syntactic parser simply by labeling individual sentences, a procedure that builds easily upon an existing workflow that already includes annotating sentences with morphological information.

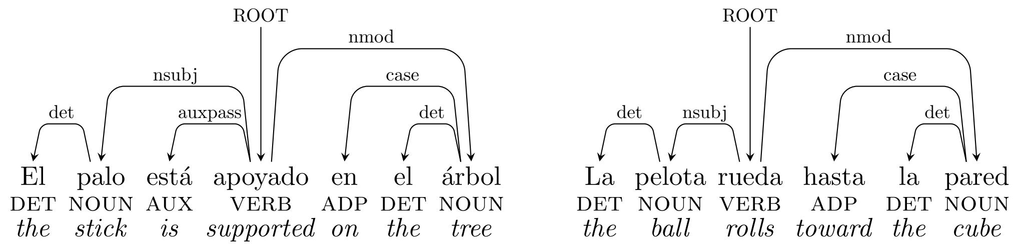
4.3 Corpus

In order to conduct the experiments described in this chapter, we needed a dependency treebank containing sentences that are similar to sentences that field linguists would probably analyze using WELT. To produce this treebank, we started with two stimulus kits used by field linguists to study spatial and motion language: the Picture Series for Positional Verbs [Ameka *et al.*, 1999] and the Motion Verb Stimulus Kit [Levinson, 2001]. For each picture and video clip, we elicited a one-sentence description from native speakers of English, Spanish, German, and Egyptian Arabic. We chose languages covering a range of linguistic phenomena. For example, German uses morphological case, and Spanish and Arabic both use clitics. In future work, we hope to add languages from other language families, including Chinese and Korean. Our languages are high-resource languages because we needed to have access to linguistically trained native speakers in order to create the gold standard; however, we did not actually use any additional resources for these languages in our experiments, and we believe they can therefore stand in for low-resource languages.

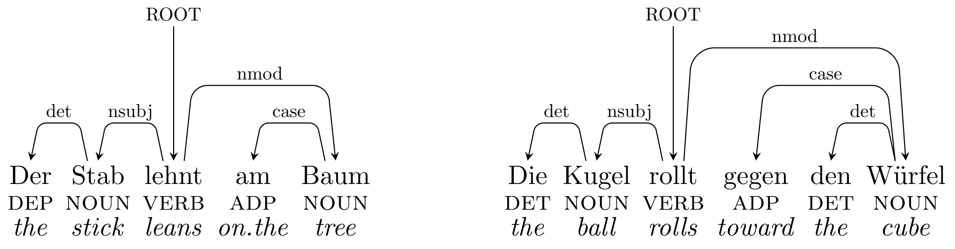
We started out by tokenizing each sentence; for Spanish and Arabic, this step included splitting off the clitics. We then annotated each token with its lemma, morphological information, part of speech, syntactic head, and dependency label. For consistency across languages, we used part of speech tags, morphological features, and dependency labels from the Universal Dependencies project [Nivre *et al.*, 2016] and attempted to follow the universal guidelines as closely as possible. The total number of sentences, average sentence length, and number of unique words, lemmas, part of speech tags, morphological features, and dependency labels for each language is shown in Table 4.1. Note that the sentence


ROOT
nmod
nsubj
case
det
aux
det
The stick is leaning against the tree
DET NOUN AUX VERB ADP DET NOUN
ROOT
nmod
case
det
nsubj
det
The ball rolls into the cube
DET NOUN VERB ADP DET NOUN

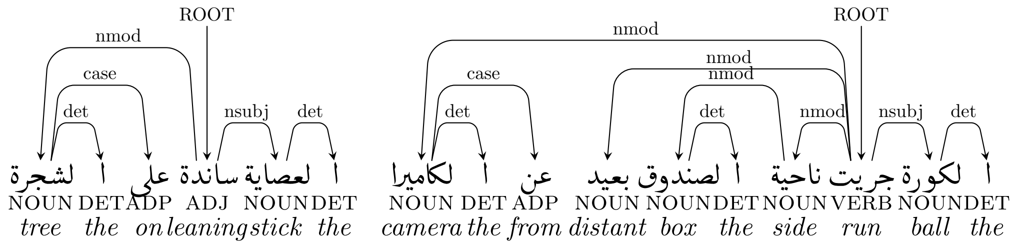
(a) English


ROOT
nmod
nsubj
case
det
auxpass
det
El palo está apoyado en el árbol
DET NOUN AUX VERB ADP DET NOUN
the stick is supported on the tree
ROOT
nmod
case
det
nsubj
det
La pelota rueda hasta la pared
DET NOUN VERB ADP DET NOUN
the ball rolls toward the cube

(b) Spanish


ROOT
nmod
det
nsubj
case
Der Stab lehnt am Baum
DEP NOUN VERB ADP NOUN
the stick leans on.the tree
ROOT
nmod
case
det
nsubj
det
Die Kugel rollt gegen den Würfel
DET NOUN VERB ADP DET NOUN
the ball rolls toward the cube

(c) German


ROOT
nmod
case
det
nsubj
det
لشجرة ا على ساندة لعصاية ا
NOUN DET ADP ADJ NOUN DET
tree the on leaning stick the
nmod
ROOT
nmod
nmod
case
det
det
nmod
nsubj
det
لكاميرا ا عن بعيد لصندوق ا ناحية جريت لكورة ا
NOUN DET ADP NOUN NOUN DET NOUN VERB NOUN DET
camera the from distant box the side run ball the

(d) Arabic

Figure 4.1: Example dependency structures

	Num. of	**Avg. sent.**			**POS**		
Language	**sents**	**length**	**Words**	**Lemmas**	**tags**	**Features**	**Labels**
English	163	7.21	152	135	12	26	20
Spanish	165	8.51	180	149	12	30	20
German	157	7.52	217	175	13	32	19
Arabic	158	10.04	174	117	10	37	15

Table 4.1: Summary of each language in the corpus. Unless otherwise specified, values indicate the count of unique types in the corpus for each category.

count varies slightly for each language; this is because for some of the pictures and videos, the native informant gave us several possible descriptions. English and German have very similar average sentence lengths; average lengths in Spanish and Arabic are higher. German had the largest vocabulary; English had the smallest vocabulary. All languages used similar numbers of part of speech tags and dependency labels, except Arabic which used fewer of both. Arabic had the largest number of morphological features, and English the smallest. Some example sentences with dependency labels are shown in Figure 4.1. The complete annotated treebank is publicly available[3].

4.4 Parsing Experiments and Results

We used four methods of training a dependency parser on our data: MSTParser [McDonald *et al.*, 2006], two configurations of MaltParser [Nivre *et al.*, 2006], and a baseline. All experiments used 5-fold cross validation. For each of the four training methods, we trained on a subset of the train fold ranging from 1 sentence to 100 sentences. We tested on the full test fold, and then averaged the accuracy across the five folds. Results are shown in Tables 4.2 and 4.3. *Arc* accuracy requires selecting the correct head for a token; *Label (Lbl)* accuracy requires selecting the correct dependency label; *Both* requires that both head and

[3]`https://doi.org/10.7916/D8W959HJ`

dependency label are correct. The highest accuracy in each row for each metric (Arc, Lbl, Both) is shown in bold.

The baseline is determined by assigning the majority dependency label from the train data. Heads are selected using left or right attachment, whichever is more common in the train data. For most of the training sets, we did left attachment and assigned det as the dependency label, and the baseline usually remained constant across all train sizes. For German with train size = 2, one of the folds had a majority of right attachment, which resulted in a slight decrease in baseline accuracy. Likewise, for Arabic with train size = 1 and train size = 2, one fold used right attachment, resulting in a decrease in arc accuracy for those rows. The baseline label accuracy for Arabic was much more variable than for the other languages, since nmod was the majority label about half of the time. The Arabic baseline used for each train size and train fold is shown in Table 4.4.

The first parser we tested was MSTParser [McDonald *et al.*, 2006, 2005], which uses a two-stage approach to parsing: an unlabeled parser and a separate edge labeler. The parser works by finding a maximum spanning tree; the label sequence is then found using Viterbi's algorithm. MSTParser uses a combination of lexical, part of speech, and morphological features; we did not modify the default feature set.

We next tested MaltParser [Nivre *et al.*, 2006], which implements a variety of deterministic parsing algorithms. A dependency structure is derived using features of the current parser state to predict the next action. Parser state is represented by a stack of partially processed tokens and a list of remaining input tokens. We tested two algorithms: Nivre arc-eager and Nivre arc-standard. The arc-eager algorithm adds arcs to the dependency tree as soon as the head and dependent are available; the arc-standard algorithm requires that the dependent already be complete with respect to its own dependents. We used the default feature sets for each of the algorithms. Like MSTParser, the feature set includes a combination of lexical, part of speech, and morphological features; MaltParser also adds dependency features (arcs and labels) from the current parser state.

Even with only one training sentence, both MSTParser and MaltParser performed well above the baseline. MaltParser consistently achieved higher accuracy than MSTParser

(a) English							MaltParser			MaltParser		
Train	Baseline			MSTParser			(Nivre arc-eager)			(Nivre arc-standard)		
size	Arc	Lbl	Both	Arc	Lbl	Both	Arc	Lbl	Both	Arc	Lbl	Both
1	0.452	0.325	0.318	0.669	0.495	0.437	0.720	0.785	0.699	**0.741**	**0.793**	**0.708**
2	0.452	0.325	0.318	0.730	0.702	0.643	0.798	**0.831**	**0.769**	**0.801**	0.826	0.764
5	0.452	0.325	0.318	0.794	0.780	0.723	**0.852**	**0.860**	**0.822**	0.817	0.843	0.789
10	0.452	0.325	0.318	0.826	0.787	0.743	**0.872**	**0.880**	**0.846**	0.829	0.856	0.804
25	0.452	0.325	0.318	0.872	0.830	0.798	**0.935**	**0.925**	**0.902**	0.878	0.901	0.855
50	0.452	0.325	0.318	0.930	0.884	0.865	**0.950**	**0.942**	**0.919**	0.920	0.925	0.897
100	0.452	0.325	0.318	0.939	0.913	0.896	**0.961**	**0.965**	**0.946**	0.945	0.951	0.926

(b) Spanish							MaltParser			MaltParser		
Train	Baseline			MSTParser			(Nivre arc-eager)			(Nivre arc-standard)		
size	Arc	Lbl	Both	Arc	Lbl	Both	Arc	Lbl	Both	Arc	Lbl	Both
1	0.397	0.273	0.271	0.454	0.329	0.311	0.504	0.570	0.478	**0.533**	**0.588**	**0.493**
2	0.397	0.273	0.271	0.568	0.444	0.397	0.600	0.650	**0.558**	**0.605**	**0.663**	**0.558**
5	0.397	0.273	0.271	0.662	0.608	0.541	**0.753**	0.779	**0.713**	0.752	**0.786**	0.702
10	0.397	0.273	0.271	0.758	0.729	0.662	0.797	0.836	0.773	**0.810**	**0.838**	**0.777**
25	0.397	0.273	0.271	0.837	0.813	0.770	**0.890**	**0.905**	**0.865**	0.868	0.887	0.843
50	0.397	0.273	0.271	0.880	0.861	0.817	**0.921**	**0.937**	**0.905**	0.910	0.930	0.895
100	0.397	0.273	0.271	0.923	0.898	0.871	**0.947**	**0.959**	**0.935**	0.932	0.947	0.919

(c) German							MaltParser			MaltParser		
Train	Baseline			MSTParser			(Nivre arc-eager)			(Nivre arc-standard)		
size	Arc	Lbl	Both	Arc	Lbl	Both	Arc	Lbl	Both	Arc	Lbl	Both
1	0.446	0.286	0.273	0.575	0.407	0.353	0.643	0.680	0.594	**0.655**	**0.685**	**0.595**
2	0.446	0.269	0.234	0.631	0.494	0.435	0.737	0.738	0.657	**0.749**	**0.770**	**0.676**
5	0.446	0.286	0.273	0.753	0.634	0.585	**0.794**	0.800	**0.732**	0.781	**0.820**	0.730
10	0.446	0.286	0.273	0.770	0.676	0.634	**0.819**	**0.848**	**0.782**	0.810	0.844	0.770
25	0.446	0.286	0.273	0.820	0.751	0.707	**0.883**	**0.896**	**0.854**	0.836	0.895	0.819
50	0.446	0.286	0.273	0.850	0.797	0.758	**0.914**	**0.936**	**0.899**	0.896	0.935	0.879
100	0.446	0.286	0.273	0.908	0.845	0.816	**0.942**	0.953	**0.931**	0.916	**0.954**	0.908

Table 4.2: Accuracy of each parsing method (to be continued in Table 4.3, for (d) Arabic). The highest accuracy in each row for each metric (Arc, Lbl, Both) is shown in bold.

(d) Arabic							MaltParser			MaltParser		
Train	Baseline			MSTParser			(Nivre arc-eager)			(Nivre arc-standard)		
size	Arc	Lbl	Both	Arc	Lbl	Both	Arc	Lbl	Both	Arc	Lbl	Both
1	0.358	0.253	0.181	0.623	0.491	0.434	**0.650**	**0.707**	**0.611**	0.617	0.681	0.571
2	0.358	0.254	0.184	**0.712**	0.656	0.598	0.704	0.738	0.646	0.687	**0.760**	**0.654**
5	0.424	0.237	0.171	0.787	0.747	0.694	0.842	0.861	0.799	**0.844**	**0.871**	**0.811**
10	0.424	0.235	0.168	0.864	0.808	0.768	0.902	0.907	0.869	**0.906**	**0.923**	**0.882**
25	0.424	0.194	0.062	0.920	0.858	0.827	**0.941**	**0.939**	**0.917**	0.930	0.938	0.909
50	0.424	0.216	0.114	0.948	0.888	0.869	0.954	0.958	0.938	**0.957**	**0.965**	**0.941**
100	0.424	0.237	0.171	0.962	0.912	0.897	**0.975**	0.972	**0.961**	0.973	**0.973**	0.957

Table 4.3: Accuracy of each parsing method (continued from Table 4.2, which shows (a) English, (b) Spanish, and (c) German). The highest accuracy in each row for each metric (Arc, Lbl, Both) is shown in bold.

for all languages and train sizes, especially when predicting the dependency labels. The performance of the arc-eager algorithm vs. the arc-standard algorithm seems to vary by language and train size. For English, Spanish, and German, the arc-standard algorithm has higher performance on small training sets, while the arc-eager algorithm becomes superior as more training data is available. Results are more mixed for the Arabic data.

4.5 Initial Experiments with Nonlexical Models

Since stimulus packs (such as the picture series and video series that we used to create our corpus) are commonly reused across many languages, it would be helpful if a parser trained on a fully-annotated version of the data for one language could be used by a field linguist just starting out with another, potentially similar, language. To that end, we performed an initial experiment to see whether a parser trained on one language could be applied successfully to the other languages in our corpus. To test this, we used MaltParser (arc-eager algorithm) to train a parser on English data, using only nonlexical features: part of speech, morphological tags, and dependency labels/arcs. We then applied this model to the

Train size	**Fold 1**	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**
1	det	det	nmod **(right)**	nmod	det
2	det	det	nmod **(right)**	det	nmod
5	det	det	det	nmod	nmod
10	nmod	det	det	det	nmod
25	nmod	det	nmod	nmod	nmod
50	nmod	det	det	nmod	nmod
100	det	det	det	nmod	nmod

Table 4.4: Arabic baseline: majority label per fold; if not otherwise indicated, the default attachment is left.

Language	**Arc**	**Label**	**Arc+Label**
Spanish	0.767	0.816	0.719
German	0.808	0.840	0.773
Arabic	0.629	0.713	0.567

Table 4.5: Accuracy of (English) nonlexical model applied to other languages

other three languages. Results are shown in Table 4.5. For this experiment, we used all 163 sentences from the English corpus.

Comparing these results to those in Tables 4.2 and 4.3, we see that, for Spanish, the English nonlexical model performs similarly to MaltParser trained on 5 Spanish sentences. For German, the English nonlexical model performs similarly to MaltParser trained on 5–10 German sentences. For Arabic, the English nonlexical model has lower accuracy than MaltParser trained on a single Arabic sentence. This suggests that a simple nonlexical model such as this one may only be useful for linguists doing this kind of annotation if an annotated corpus in a *related* language is available.

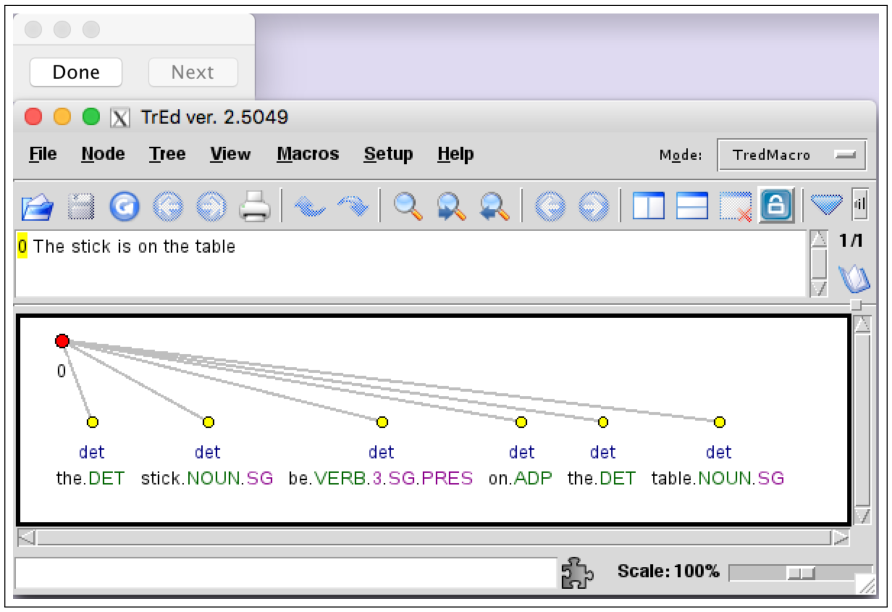
4.6 Annotation Experiments and Results

To test whether a parser trained in this manner would help with annotation, we performed annotation experiments using the English data. We timed how long it took an annotator to label a sentence, when the sentence is preprocessed in one of four ways. In the first method, a baseline assigns a flat structure and the dependency label `det` to all nodes. The other methods use MaltParser (Nivre arc-eager algorithm) trained on 1, 5, or 25 sentences to provide an initial parse for the annotator to correct.

Annotators labeled five trees for each parsing method, for a total of 20 trees. To ensure each of the four sets of five contained sentences with similar syntactic complexity, the sentences were chosen as follows. Each parsing method was assigned one sentence of each of five lengths: 7 words, 8 words, 9 words, 10–11 words, and 12–14 words. These were randomly selected from among all sentences of the required length. The 20 sentences were then presented to the annotator in random order. To keep the experiment consistent, all annotators labeled the same 20 sentences, in the same order.

Three annotators participated in the experiment. The first was the author of this thesis. She is an expert annotator, very familiar with the universal guidelines for dependency annotation and the annotation software. The other two annotators were undergraduate students who participated in a brief training session to familiarize them with the desired analysis and the software. They were given reference materials showing sample trees covering a variety of syntactic phenomena including: auxiliaries, copulas, coordination, secondary predication, and subordinate clauses. They were also able to refer to this material throughout the annotation task. Before participating in the annotation task, they annotated 10 additional trees for practice.

The software used for annotation was Tree Editor (TrEd) [Pajas and Štěpánek, 2008] with a simple Java wrapper that handled opening files in TrEd and keeping track of annotation time. Upon pressing the "Next" button, the annotator was shown the next tree in TrEd and the program recorded the start time. When the annotator finished labeling a tree, they saved the file in TrEd and pressed the Done button. The wrapper program closed the current file in TrEd and recorded the end time. Figure 4.2 shows a screenshot of the



0 The stick is on the table

0

det det det det det det

the.DET stick.NOUN.SG be.VERB.3.SG.PRES on.ADP the.DET table.NOUN.SG

Figure 4.2: Screenshot of software used for annotation experiments; the sentence shown here was pre-processed using the baseline method.

setup for a sentence that was pre-processed using the baseline method.

A detailed listing of the time taken by each annotator to label each sentence size is shown in Table 4.6. The graph in Figure 4.3 shows the average time (across all annotators) taken to label each sentence size when using each parsing method. The graph in Figure 4.4 shows the average time (across all sentence lengths) taken by each annotator to label a sentence when using each of the four parsing methods. All times are given in seconds. A graph showing the accuracy of the baseline and of MaltParser (arc-eager, with train size of 1, 5, and 25) on sentences of different lengths is shown in Figure 4.5.

Results vary slightly across annotators, but it is clear that, even when training on a single sentence, annotation time is improved. Average annotation time decreases from 104.5 seconds for the baseline parse to 79.9 seconds for the parser trained on one sentence. Using the parser trained on 5 sentences, average annotation time decreases again to 69.6 seconds. Using the parser trained on 25 sentences, we see a decrease in annotation time to an average of 45 seconds. Statistical significance testing was done with a paired t-test. Significant decreases in annotation time are: between the baseline and 1 training sentence

(a) Student1

Sent. size	Baseline	Malt-1	Malt-5	Malt-25
7	87.0	54.9	14.4	14.3
8	73.0	83.6	65.1	17.7
9	85.9	67.2	61.7	43.3
10-11	144.3	70.6	65.7	86.6
12-14	104.5	107.5	237.1	57.3
Avg	98.9	76.8	88.8	43.8

(b) Student2

Sent. size	Baseline	Malt-1	Malt-5	Malt-25
7	132.7	54.0	23.6	26.3
8	97.8	98.9	24.3	30.4
9	90.8	153.0	96.9	102.4
10-11	203.4	66.2	139.9	138.9
12-14	240.8	274.1	203.7	68.7
Avg	153.1	129.2	97.7	73.4

(c) Expert

Sent. size	Baseline	Malt-1	Malt-5	Malt-25
7	59.2	25.4	8.1	8.7
8	44.7	42.0	8.6	9.2
9	52.3	29.7	21.4	21.0
10-11	75.8	34.9	11.2	36.8
12-14	75.5	36.0	62.0	12.8
Avg	61.5	33.6	22.3	17.7

(d) Average

Sent. size	Baseline	Malt-1	Malt-5	Malt-25
7	93.0	44.8	15.4	16.5
8	71.8	74.8	32.7	19.1
9	76.3	83.3	60.0	55.6
10-11	141.1	57.3	72.3	87.5
12-14	140.3	139.2	167.6	46.3
Avg	104.5	79.9	69.6	45.0

Table 4.6: Time (seconds) for an annotator to label a sentence, for each parsing method and sentence size.

($p = 0.034$), between the baseline and 5 training sentences ($p = 0.015$), between the baseline and 25 training sentences ($p = 2.51\text{e}{-5}$), and between 1 training sentence and 25 training sentences ($p = 0.018$).

There are several explanations to account for the fact that training on a single sentence significantly decreases annotation time. MaltParser works by predicting steps in a derivation, so one sentence actually translates into more than one data point. With only one sentence, the parser can learn that a determiner should be the left child of a noun, or that a noun should be the left child of the root predicate. Having these dependencies already

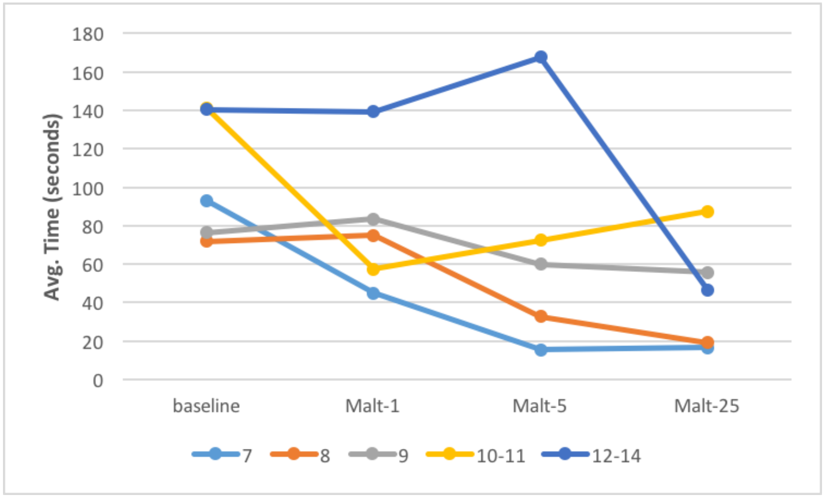


Figure 4.3: Average annotation time for each sentence size, across all annotators

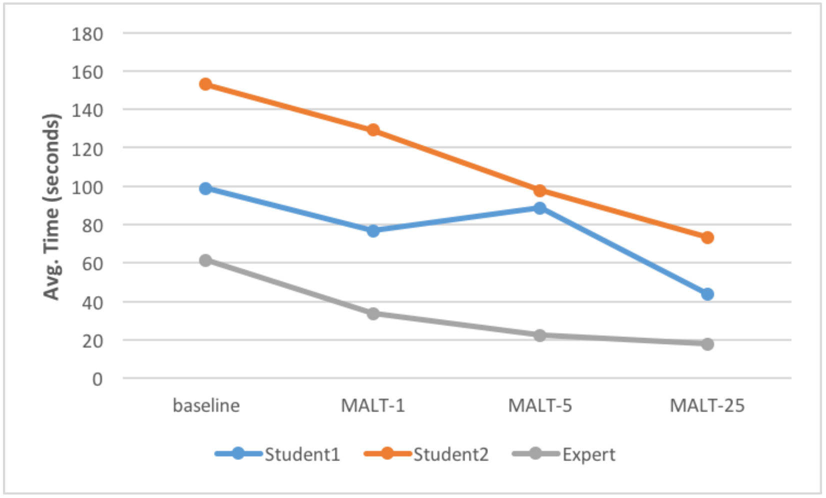


Figure 4.4: Average annotation time for each annotator, across all sentence sizes

attached reduces the work the annotator must do compared to a completely flat structure. In addition, our corpus consists only of descriptions of spatial relations and motion events, so we expect a much more limited range of grammatical constructs than one finds in other treebanks.

For very short sentences (length 7), the graph in Figure 4.5 shows a clear downward trend as the amount of training data increases. For sentences of length 8–9, we do not see

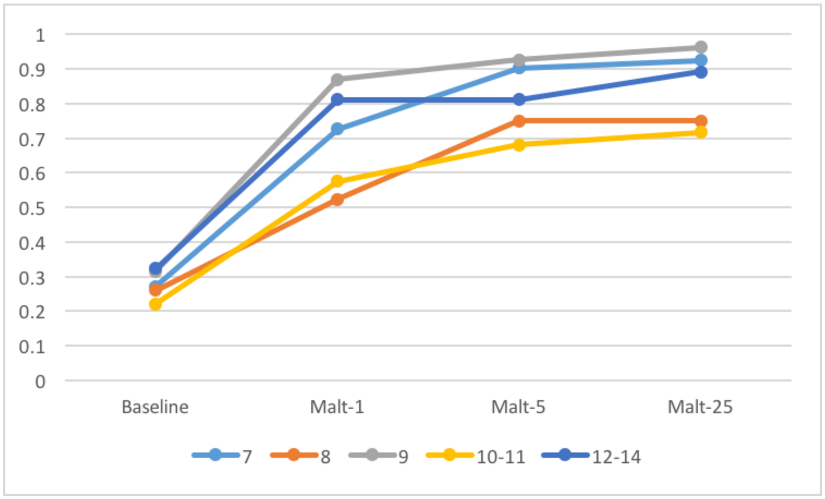


Figure 4.5: Parser accuracy on each sentence size, for baseline method and for MaltParser (arc-eager) trained on 1, 5, or 25 sentences.

improvement with one training sentence, but annotation time begins to decrease substantially when there are 5 training sentences. For longer sentences, the downward trend is less clear. This makes sense, since we can expect to find a wider range of syntactic structures in a longer sentence, and parser performance on these will require that a similar structure was seen in the train set. For sentences of length 10–11, there is a substantial drop in annotation time from baseline to 1 training sentence, at which point it seems to plateau. For sentences of length 12–14, average annotation time does not decrease until we have 25 training sentences.

One concern with using a parser to assist with annotation is whether there will be any effect on overall accuracy. When presented with a mostly-correct parse, will annotators be able to see all the errors and fix them? The accuracy of the annotators for each of the four parsing types is shown in Table 4.7. We do see a drop in accuracy for all annotators when training on 5 sentences, especially for Student1. However, this decrease is very slight for both Student2 and Expert. We suspect there may have been several difficult sentences in this set; all annotators made errors on the sentence of length 10, and Student1 had particularly low accuracy (0.625) on the sentence of length 8.

Parser	**Student1**			**Student2**			**Expert**		
	Arc	**Lbl**	**Both**	**Arc**	**Lbl**	**Both**	**Arc**	**Lbl**	**Both**
Baseline	0.951	1.000	0.951	1.000	1.000	1.000	1.000	1.000	1.000
Malt-1	0.971	1.000	0.971	1.000	1.000	1.000	1.000	1.000	1.000
Malt-5	0.930	0.950	0.905	0.980	1.000	0.980	0.980	1.000	0.980
Malt-25	0.962	0.982	0.962	1.000	1.000	1.000	1.000	1.000	1.000

Table 4.7: Annotation accuracy for each annotator, across all sentence sizes

4.7 Conclusion

We have reported results for incrementally training a dependency parser across four languages. Our results show that such a parser can improve on baseline performance even when trained on a single sentence, making our method particularly useful in the documentation of endangered and low-resource languages. We found that MaltParser achieved the highest accuracy overall; the arc-standard algorithm seems preferable for very small training sizes and arc-eager for slightly larger training sizes. We found that using a parser to predict each sentence before annotation did significantly improve annotation time, without a substantial decrease in accuracy.

The results of our experiments demonstrate that it is possible to extend FieldWorks' "stealthy" approach to learning a morphological parser into the realm of syntax. The first goal for our future work on this topic would be to incorporate these methods into WELT, by providing an interface for specifying the syntax of sentences in the form of dependency structures and using them to train a parser in the background. The parser would provide predictions for new sentences, and, as these are corrected and approved by the linguist, they would be added to the training data and the parser is incrementally improved. By providing a way to assign dependency structures to sentences, this will allow field linguists to incorporate syntax into language documentation. The incrementally trained parser will reduce their workload by letting them correct errors in a dependency structure rather than starting from scratch. This method of syntactic documentation does not limit the field linguist to a particular syntactic theory. We chose to use the universal labels and analyses in our corpus, but parsers could be just as easily trained using any other theory for head assignment and choice of dependency labels, as long as they are consistent across sentences.

Another avenue for future work on this topic is to experiment with other parsers, such as TurboParser [Martins *et al.*, 2010], Mate [Bohnet, 2010], and Easy-First [Goldberg and Elhadad, 2010]. In addition, one could continue to investigate methods of re-using existing parsers and dependency annotations with new languages (see Section 4.5); specifically, to investigate more effective methods of adapting existing parsers to other languages. For example, one could investigate how to combine a non-lexical model with a lexical model obtained from a small number of target language sentences. Another area for future work is to investigate ways for linguists to directly specify syntactic properties that can be used by the parser, similar to the way FLEx converts morphological properties specified by users into formal rules compatible with the underlying parser.

Chapter 5

3D-Scene Generation from Semantic Primitives

5.1 Introduction

In this chapter, we will describe how we have adapted WordsEye into a novel system that generates a 3D scene from semantic primitives representing basic spatial and graphical relations. The motivation for this system is twofold. First, although it is true that generating scenes from natural language can simplify the process of creating of custom 3D content, there are situations when a natural language interface is not ideal. Some issues associated with using natural language input, both in general and specifically in our target applications, are:

- 3D scenes created from natural language are limited by errors the system makes processing and parsing the language of the input text. Even in the absence of system errors, it can be difficult to achieve a precise spatial configuration due to the ambiguity of natural language.[1]

[1] We saw examples of this when used WELT to create 3D scenes representing topological relations for eliciting Nahuatl (Chapter 3). The 3D scenes we created, along with the input text used to generate them, are included in Appendix B. While in many cases using English as input did simplify the process of creating our scenes, in other cases the reliance both on English and on the English language models used by WordsEye made it necessary for us to construct highly contrived and lengthy input text.

- A natural language interface means that users must be competent in the input language (usually English).
- Using English to create scenes that will be used for elicitation in another language may introduce unwanted biases and assumptions about the target language.

The second motivation is based on the WELT documentation tools. The WELT documentation tools let linguists formally document the semantics of a language by specifying syntax-to-semantics rules. These rules are mapped into WordsEye vignettes, a higher-level semantics that is often both specific to the English language and also biased toward Western culture. In Chapter 3, we addressed this issue by extending VigNet with custom L2 vignettes and 3D content. However, editing VigNet at this level currently requires manual intervention by WordsEye developers to edit the system and patch the desktop program. The ability to generate 3D scenes directly from spatial and graphical primitives will allow us to remove the dependence of the semantic documentation on English-specific parts of the WordsEye system, ensuring that the mapping can always be to a language-independent semantics. In Chapter 6, we will discuss how the spatial and graphical primitives can be used as a semantic grounding for vignettes in other languages, independently of the English version VigNet used by WordsEye.

We address these problems by introducing a system that generates a 3D scene from semantic primitives representing basic spatial relations and graphical properties. To do this, we divide the text-to-scene generation problem into two sub-tasks: (a) conversion of input text into an underlying semantic representation, and (b) conversion of that semantic representation into a 3D scene. In this way, we separate the *language-dependent* task of converting input text into semantics from the *language-independent* task of converting a semantic representation into a 3D scene. Our system gives users direct access to view and modify the underlying semantics of the 3D scene being created instead of relying on black-box conversion from input text to rendered scene. Users can provide the system with English textual input and subsequently edit the semantic representation, or they can directly provide the desired semantics for the scene.

This chapter is organized as follows. In Section 5.2, we will discuss related work. In

Section 5.3, we describe adapting WordsEye to allow direct access to the underlying semantics of a generated scene, including (a) the ability to view and edit the semantics of a generated scene, and (b) the ability to generate 3D scenes directly from graphical primitives. We include an evaluation of the semantic output produced by WordsEye. In Section 5.4, we describe incorporating this new functionality into the WordsEye Linguistics Tools, improving the user interface that allows field linguists to create custom elicitation materials in the form of 3D scenes. We conclude in Section 5.5.

5.2 Related Work

Other systems exist for producing graphics from natural language sources. Some of these are discussed in Section 2.2. In most existing systems, the referenced objects, attributes, and actions are relatively small in number or targeted to specific pre-existing domains. WordsEye, the text-to-scene system used in this thesis, was one of the first text-to-scene systems designed for general use rather than a specific domain.

The main difference between our work and previous work in text-to-scene generation is the distinction we make between *language-dependent* and *language-independent* capacities of the system. Other text-to-scene systems use intermediate representations between input text and graphics generation; Chang [2015] gives an overview of some of these. However, even systems using semantics as an intermediate representation often conflate the *semantic concepts* of spatial relations with the *lexical items* that represent those concepts. When Chang *et al.* [2014b] describe learning spatial knowledge from crowdsourced data, the spatial relations learned are in fact keywords extracted directly from English text. WordsEye also has a semantic component tied to English. Semantic analysis in WordsEye relies on VigNet [Coyne *et al.*, 2011a], a lexical, semantic, and graphical resource. Many of the conceptual and graphical relations in VigNet are based on English lexical items, and the English-specific portions of VigNet are not easily separated from the language-independent graphical and world knowledge.

Another type of graphics-generation system takes as input a set of high-level constraints rather than in natural language. This approach is another way of allowing users to create 3D scenes without expert knowledge of graphics. Some of these systems use a domain-specific design language as input and others allow users to specify constraints in a GUI. Plemenos and Miaoulis [2009] provide a survey of some systems and techniques. Our work is similar since semantic input to WordsEye can be viewed as a set of constraints on the 3D scene. However, we support both natural language and semantic constraints as input.

As we described in Chapter 3, WordsEye has previously been used to support field linguistics, as part of the WordsEye Linguistics Tools, which consists of tools for elicitation and documentation of an endangered language. The work in this chapter improves on the WELT elicitation tools in two ways. First, we use the latest version of WordsEye, interacting directly with the current WordsEye web API instead of the earlier desktop program. Second, we add semantic input functionality, which allows users more precise control over the generated scene.

5.3 Modification of WordsEye

5.3.1 Adding Semantic Input and Output

Figure 5.1 shows the WordsEye architecture with the new semantic input and output modules introduced in this chapter. To add this functionality, we worked with the WordsEye developers to implement the necessary changes to the WordsEye system. WordsEye provides a web API, previously used to develop an iPhone app. The API takes text and other information (such as preferred camera position) as input. WordsEye processes the text, then returns a JSON object that includes a JPEG image of the rendered scene and additional information (such as choices of 3D objects). WordsEye creates an intermediate semantic representation of the scene as part of this process (see Chapter 2, Section 2.3). The API was modified to accept a semantic representation directly, bypassing normal text-processing modules. The output was modified to return the semantic representation along with the rendered scene. In addition, parts of the VigNet resource (relations, ontology, and assertions) were converted into JSON format so they can be referenced by external applications.

The semantic representation is specified by a JSON object and consists of a set of variables representing entities in the scene and a set of relations between those entities.

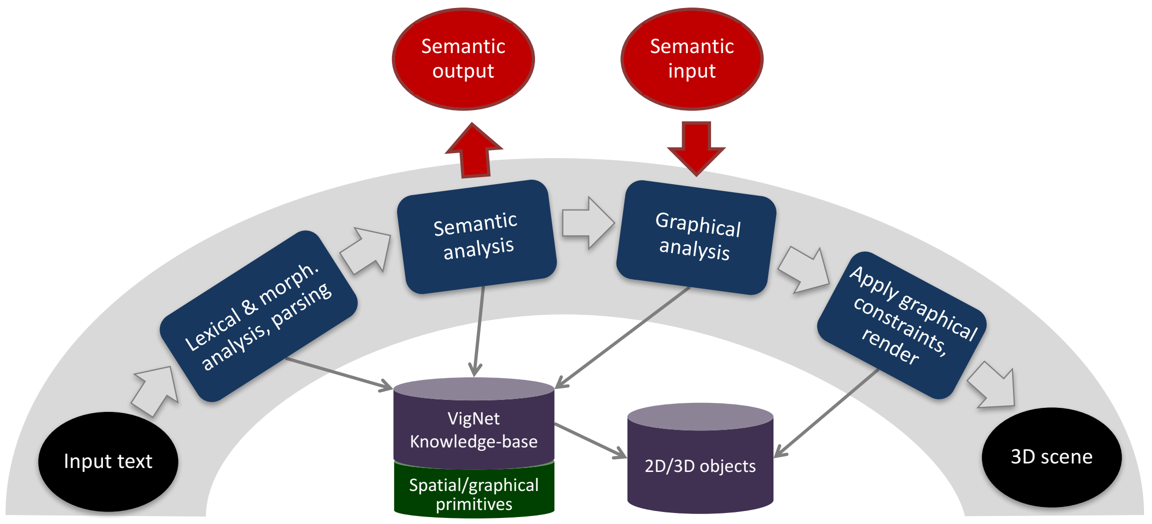


Figure 5.1: WordsEye system architecture. We have modified the pipeline to add semantic output and input modules (see Section 5.3.1). We have also separated primitive spatial and graphical relations from other relations in VigNet (see Section 5.3.2).

Arguments of relations are filled by entity variables and semantic concepts from the VigNet ontology. Semantic output and input use the same format, so the JSON object returned by the WordsEye API can also be used as input to the system. Semantic output for *The striped rug is in front of the sofa* is shown here:

```
{"entities": [
    {"id":"n.1", "isa":["couch.n"] },
    {"id":"n.2", "isa":["rug.n"] },
    {"id":"n.3", "isa":["stripe.n"] }
  ],
 "relations": [
    {"id":"r.1",
     "isa":"gfx.in-front-of.r",
     "core-args": {"ground":"n.1", "figure":"n.2"}
    },
    {"id":"r.2",
     "isa":"gfx.has-texture.r",
     "core-args": {"entity":"n.2", "texture":"n.3"}
    }
  ]
}
```

Although the WordsEye system uses an intermediate semantic representation as part of its normal processing pipeline, we found that this representation required additional modification before it could be used in our application. The first example of changes that were necessary occurs because WordsEye does not always completely disambiguate the input text before producing its intermediate representation. Sometimes initial text processing results in several possible semantic concepts for a lexical item, and the final disambiguation isn't made until the graphical analysis stage of the pipeline. For example, in *the dog on the wood chair*, the intermediate representation in WordsEye contains both WOOD.N (referring to the substance) and PIECE-OF-WOOD.N as possible interpretations of *wood*. In this case, we are describing the texture of the chair, so the first option is the correct interpretation. This determination is made by WordsEye when it is constructing the graphics for the scene. WordsEye does this by choosing the concept that satisfies the constraints of the given graphical primitive. For example, the GFX.TEXTURE-OF.R relation will accept a image or substance but not a 3D object for the texture it applies. We wanted the semantic representation to include only the appropriate concept for each lexical item, so it was necessary to add additional processing of the internal semantic representation before it is returned by the API, to add information obtained during graphical analysis.

Another example is that for a given input text, there can be several ways to specify the underlying semantics. The representation used internally by WordsEye is not always the most intuitive one for human users. For instance, two possible semantic representations (in JSON format) for the sentence *the bottle is in the basket* are shown in Figure 5.2. The most obvious representation (Figure 5.2, left) would be to use the GFX.IN-CUP.R relation between FIGURE (the bottle) and GROUND (the basket). Instead, WordsEye uses an alternative representation (Figure 5.2, right) in which the relevant part of the basket, the cup region, is specifically referenced. In addition to the bottle and the basket, an entity with semantic type CUP-TAG.N is included in the semantic representation. Then, three relations are used to specify the spatial configuration. The new entity is specified to be a spatial tag with GFX.SPATIAL-TAG.R. This spatial tag is specified to be a part of the basket with WE.PART-OF.R. Finally, GFX.ON-TOP-SURFACE.R is used for the relation between the bottle and the spatial tag. Both semantic representations will result in the same graphical configuration;

```
{"entities": [
    {"id":"n.1",
        "isa":["basket.n"]
    },
    {"id":"n.2",
        "isa":["bottle.n"]
    }
 ],
 "relations": [
    {"id":"r.1",
     "isa":"gfx.in-cup.r",
     "core-args": {
        "ground":"n.1",
        "figure":"n.2"
     }
   }
 ]
}
```

```
{"entities": [
    {"id":"n.1", "isa":["basket.n"]},
    {"id":"n.2", "isa":["bottle.n"]},
    {"id":"n.3", "isa":["cup-tag.n"]}
 ],
 "relations": [
    {"id":"r.1",
     "isa":"gfx.spatial-tag.r",
     "core-args": {"object":"n.1",
                   "tag-type":"n.3"}
    },
    {"id":"r.2",
     "isa":"we.part-of.r",
     "core-args": {"whole":"n.1",
                    "part":"n.3"}
    },
    {"id":"r.3",
     "isa":"gfx.on-top-surface.r",
     "core-args": {"ground":"n.3",
                    "figure":"n.2"}
    }
  ]
}
```

Figure 5.2: Two possible semantic representations for *the bottle is in the basket.*

however, when a human user is working directly with the semantics, the first and more simple representation is preferable. In fact, when processing text, WordsEye first produces a higher-level relation like GFX.IN-CUP.R and subsequently decomposes it into a lower-level representation using spatial tags in order to construct the graphics of a scene. To retain the higher-level relation as part of the semantic representation, we modified the the system to defer semantics decomposition into its lowest level to a later stage, so that it is done as part of its graphics analysis rather than when constructing the semantics.

WordsEye continues to support both levels of representation in the graphics module, so either may be used in semantic input. If a GFX.IN-CUP.R relation is specified in the input, the graphics module will find a cupped region on the GROUND to put the FIGURE in that position. If, instead, a specific region or part on the GROUND is specified directly in the input and a GFX.ON-TOP-SURFACE.R relation is used, then the graphics module can avoid having to find the region first. We note that in some cases it is necessary to specify a region or part – in particular when there are many possible non-default parts to choose from. But in the common case, there is usually one default region for putting something "in" or "on," and that will be be used automatically. If there is no known default region of that type (as declared by the VigNet knowledge base), then a more primitive spatial relation will be used. For "on", it will use the bounding box of the ground. For "in" it will embed the FIGURE in the GROUND rather than putting it on a containing or supporting surface.

5.3.2 Spatial and Graphical Primitives

Because one of our goals is to use the WordsEye semantic interface with languages other than English, we wanted to ensure that our system supports a language-independent semantic representation. VigNet contains a wide range of semantic relations, from high-level abstract relations originating in FrameNet, such as FN.ABANDONMENT.R, to low-level graphical relations, such as GFX.RGB-VALUE-OF.R. We extracted from VigNet a list of relations representing basic spatial configurations and graphical properties, separating these from the higher-level relations in VigNet which may be English-specific. Since these relations are closely tied to the graphics of constructing a scene rather than the English input text, a semantic representation using them will be language-independent. The list includes about 100 primitive relations that can currently be used in semantic output/input and is included in Appendix D. Although any relation in VigNet can be used in the semantic input and output, in practice, the semantic output usually includes only relations from this set of primitives. This is because the intermediate semantic representation used internally by WordsEye is intended to be translated directly into a graphical configuration.

We will return to the discussion of these primitives in Chapter 6, Section 6.4.2, where they will provide the basis for the formal set of *spatio-graphic primitives* used by SpatialNet.

5.3.3 Evaluation of Semantic Output

To evaluate the accuracy of the semantic output produced by WordsEye, we use the English descriptions of the Topological Relations Picture Series [Bowerman and Pederson, 1992] that were collected by Werning [2016]. Each description denotes a basic spatial configuration

	Relation	**Figure**	**Ground**
Correct	34 (47.2%)	67 (93.1%)	67 (93.1%)
Partial	19 (26.4%)	2 (2.8%)	0
Incorrect	19 (26.4%)	3 (4.2%)	5 (6.9%)
Total	72 (100%)	72 (100%)	72 (100%)

Table 5.1: Results of evaluation of semantic output

containing two objects: a figure and a ground. We used the WordsEye API to get the semantic representation for each description. We then checked the accuracy of the spatial relation, the figure, and the ground, each of which is classified as one of the following: completely correct, completely incorrect, or partially correct. We consider a relation or entity partially correct if it is a possible interpretation of the input sentence, but not the most likely interpretation. For example, for the sentence *The ring is on the finger*, the spatial relation in the semantic output was GFX.ON-TOP-SURFACE.R. While it is technically possible that the ring is balanced on top of a finger, the more likely interpretation is to have a finger be inserted through the ring. Another example is for the sentence *The water hose is lying on top of the stump*, where we might expect *water hose* to result in an entity with semantic type such as GARDEN-HOSE.N. Instead, WordsEye returns a generic HOSE.N entity with a WATER.N texture, using GFX.HAS-TEXTURE.R.

The evaluation results are shown in Table 5.1. In almost all cases, the semantic output contained the correct semantic types for both figure and ground. Most of the incorrect relations were due to the presence of verbs in the input text, since WordsEye currently supports only a few verbs. In this case, the semantic output usually contains a FrameNet relation matching the lexical item, and retains syntactic dependencies rather than semantic arguments. For example, for *The picture is hanging on the wall*, WordsEye returns FNEW.PATH-SHAPE.HANG.R with a dependent "on" and WALL.N as object-of-prep. In many cases like this, WordsEye would have correctly interpreted the input if the sentence structure had been simplified. For instance, input of *The picture is on the wall* correctly returns GFX.ON-FRONT-SURFACE.R.

Note that an accurate semantic representation does not guarantee that WordsEye will generate the desired graphical representation. This could be due to not having a 3D object for a given concept in the ontology. When WordsEye encounters a concept without a corresponding 3D object, it attempts to choose a related concept based on relationships in the ontology. It could also be due to graphical limitations of the system. For example, WordsEye currently cannot graphically represent a FITTED-ON relation (e.g., a hat on a head or a glove on a hand). When WordsEye encounters a relation that it cannot decompose into supported graphical primitives, the relation is ignored and not included in the 3D graphics. The entities referenced by the relations will be displayed in a default position (side-by-side). The API will also return a warning or error message in the JSON structure when it encounters anything it cannot handle.

5.4 Application to Elicitation in Field Linguistics

In this section, we describe how we use our new system to improve the elicitation tools provided by WELT English, which were introduced in Chapter 3. WELT English provides an interface for linguists to create custom elicitation materials by inputting English text into the desktop application version of WordsEye. Elicitation sessions are organized around sets of 3D scenes; the linguist elicits descriptions of these scenes from a native speaker of an endangered language.

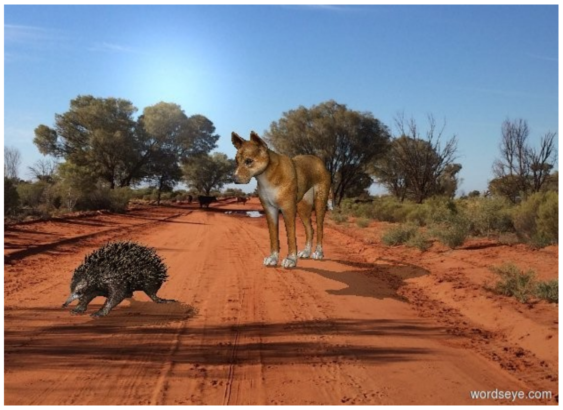
We focus here on the tools for creating the 3D scenes which will be used for elicitation. The first improvement we make is to use the latest version of WordsEye, now a platform-independent web application. The earlier desktop version of WordsEye is for Mac OS X only and requires all 3D objects used by the program (about 60 GB) to be stored on the user's computer. Since WordsEye does not own the 3D objects, users must acquire an expensive license from another company to get access to the object files. In addition, since the desktop program is no longer supported by the WordsEye developers, any bugs or other issues that come up may not be addressed. In contrast, the WordsEye web application requires no local storage or configuration, and users do not need a license for the 3D objects because these are stored on the WordsEye servers. Additionally, by interacting directly with the

WordsEye web API, we benefit from bug fixes and other updates automatically. The latest version of WordsEye also incorporates a significantly enhanced set of 3D objects and 2D images, and it uses GPU-based ray-tracing to produce visually richer scenes with shadows, reflections, and transparency (including refraction through glass).

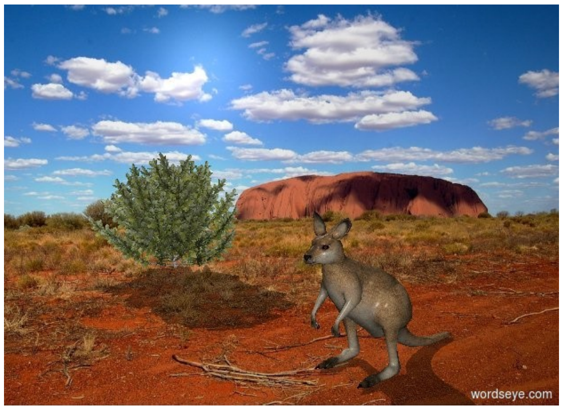
Note that along with the benefits of using the WordsEye online API, there are some downsides. First, not all the functionality provided in the desktop application has been implemented in the web application at this time. This includes putting characters in different poses and the ability to add bounding boxes to the objects in the scene. In addition, using the web API to access WordsEye requires that the user have access to an internet connection, which may not always be the case for linguists working with speakers of endangered languages in remote locations.

A key contribution of WELT is that it allows users to create elicitation materials that are culturally and geographically relevant to the speakers of an endangered language. For example, we created a collection of 2D images that would be particularly relevant to speakers of Arrernte, an Australian language spoken in and around Alice Springs. In order to incorporate these into the desktop version of WordsEye, however, the images had to be manually added by the WordsEye developers and patched into the application. The web application version of WordsEye allows users to upload their own custom images, which can be incorporated into scenes either as 2D cutouts or as backdrops. Backdrops are another new feature in the latest version of WordsEye; they are 2D images that simulate 3D environments. While it is possible to create a 3D model or scene that simulates a real-world location, it is often much easier to use a photo or illustration as a backdrop. The key constraint on backdrops is that they must have a sizeable cleared area that functions as a ground plane where 3D scene objects can be placed. When WordsEye renders 3D objects on top of a backdrop, it casts shadows onto the ground plane so as to "anchor" the 3D objects into the overall scene. WordsEye currently has about 1500 backdrops, with more being added regularly. Figure 5.3 demonstrates how backdrops can be used to create scenes that reflect the geography of the Northern Territory of Australia.

The second improvement we make is to enable editing the underlying semantics of a scene. We have created an intuitive interface that lets users enter English text, convert this



Input text: *The outback backdrop. The dingo is 2 feet behind the echidna.*

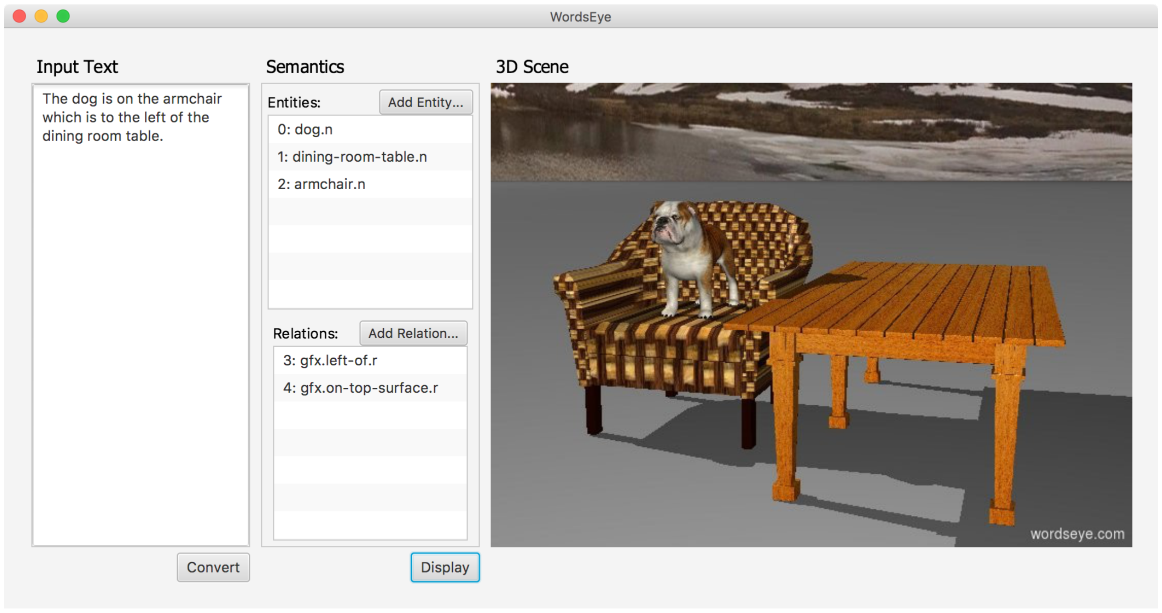


Input text: *The outback backdrop. The kangaroo is 6 feet to the right of the bush.*

Figure 5.3: Using WordsEye backdrops to create scenes that are relevant to endangered language speakers

text to semantics, optionally edit the semantics, and finally generate a 3D scene. Figure 5.4 shows a screenshot of the user interface. Input text provided by the user is converted to a JSON object representing the semantics as described in Section 5.3.1. We display this information in our user interface as lists of entities and relations. Each entity and relation in the semantic representation is displayed by its ID along with the type provided by the `isa` entry in the JSON output. For entities, this is a semantic type restriction. For relations, this is the name of the primitive graphical relation. Users can add, remove, and edit entities and relations in the semantics. Semantic types of entities can be changed to other concepts from the VigNet ontology. Arguments of relations can be modified to refer to other entities and semantic concepts. Once the user is satisfied with the semantics, the program converts it back to a JSON representation, sends it as input into the modified WordsEye API, and displays the resulting 3D scene.

The ability to edit the intermediate semantics provides several benefits to users. First, it allows much more precise control over the scene that is displayed. Instead of relying on WordsEye's interpretation of a given noun, the user can select entities and objects directly from the VigNet ontology. Similarly, the user can select a precise graphical relation. This is helpful when constructing a scene with a non-typical layout of objects. For example, given

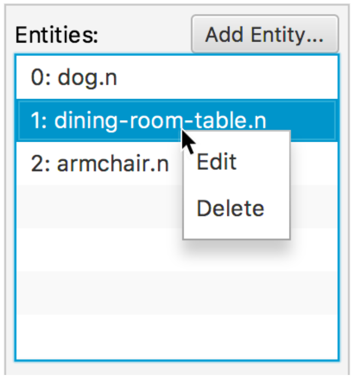


The dog is on the armchair which is to the left of the dining room table.

0: dog.n
1: dining-room-table.n
2: armchair.n

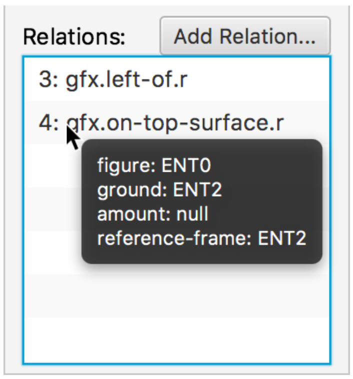
3: gfx.left-of.r
4: gfx.on-top-surface.r

(a)



0: dog.n
1: dining-room-table.n
2: armchair.n

(b)



3: gfx.left-of.r
4: gfx.on-top-surface.r

figure: ENT0
ground: ENT2
amount: null
reference-frame: ENT2

(c)

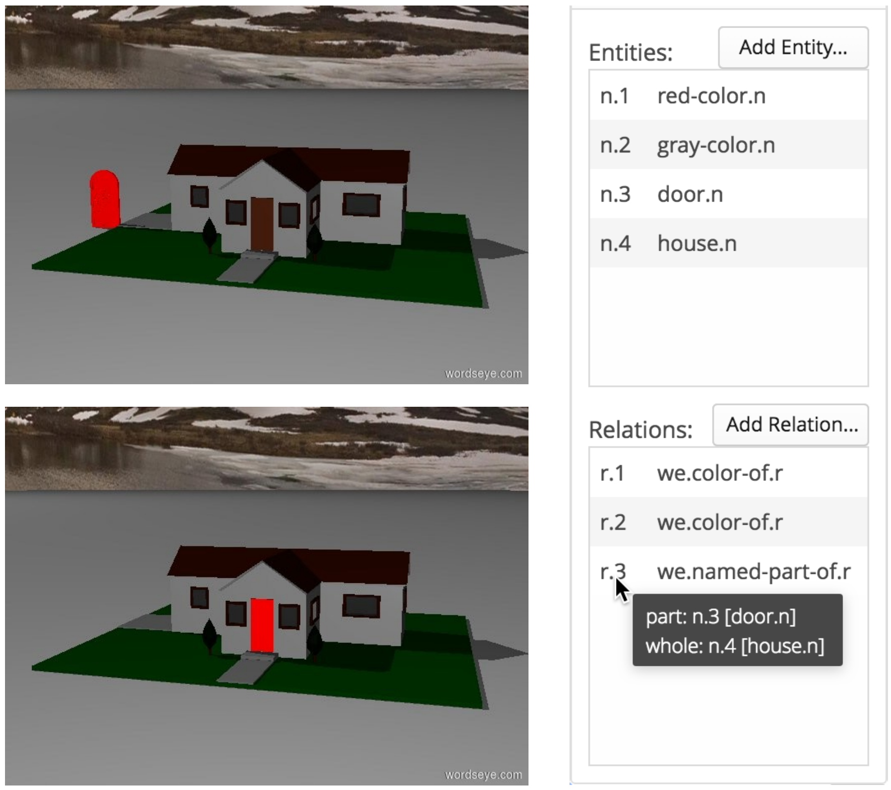
Figure 5.4: (a) Screenshot of the new user interface for 3D-scene generation in our elicitation tool. Users can view the underlying semantics and modify the generated scene by adding, removing, and/or editing entities and relations. (b) Right-clicking on an entity or relation brings up a context menu with options to edit or delete. (c) Hovering over an entity or relation reveals more information, such as values for a relation's arguments.

the input text *The painting is on the wall*, WordsEye uses the information in VigNet to find that (a) PAINTING.N is a WALL-ITEM, and (b) WALL.N has a preferred surface of FRONT-SURFACE.N. It infers from this that the relation GFX.ON-FRONT-SURFACE.R. If the user instead wants the painting to be balanced on top of a wall, they can use our user interface to change the relation to GFX.ON-TOP-SURFACE.R, keeping the argument values the same. This precise targeting of specific entities and relations is also helpful for a linguist working in real-time with an informant. Often, one wants to change a single thing in a spatial layout while keeping everything else the same, to see the effect on the informant's description. This could be changing the spatial relation between two objects, or changing one of the objects participating in the relation. With the ability to modify the semantics directly, the user can change a single item in the scene without re-processing text through WordsEye, which might result in additional changes.

Being able to work directly with the semantics also means the user is not limited by processing errors WordsEye makes during the conversion from text to semantics. For example, in the evaluation of the WordsEye system described in Ulinski *et al.* [2018], one of the system's errors was for the input text *a gray house with a red door*. The problem is that WordsEye does not interpret *with* in this sentence as signifying that the red door is part of the gray house; instead, WordsEye shows the door and house as separate 3D objects. This error is easy to correct, given access to the semantics of the scene. The initial semantics produced by WordsEye already includes the correct WE.COLOR-OF.R relations; the user just needs to add a WE.PART-OF.R relation between the door and the house to generate the intended scene. Figure 5.5 shows the original graphics produced by WordsEye, the corrected semantics, and the scene that is generated from the corrected semantics.

5.5 Conclusion

We have described the adaptation of a text-to-scene system to (a) generate a semantic representation from input text and (b) convert a semantic representation to a 3D scene. We described our use of the system to improve the WordsEye Linguistics Tools. In Chapter 6, we we will describe how we use the system as part of the semantic grounding for SpatialNet,



n.1	red-color.n
n.2	gray-color.n
n.3	door.n
n.4	house.n

r.1	we.color-of.r
r.2	we.color-of.r
r.3	we.named-part-of.r

part: n.3 [door.n]
whole: n.4 [house.n]

Figure 5.5: Using semantic output/input to fix errors in WordsEye's processing of *a gray house with a red door*. The original scene (top left) is corrected by adding the part-whole relation to the semantic representation (right), which is passed back to the WordsEye API to generate the correct scene (bottom left).

a novel resource which provides a new cross-linguistic framework for defining the lexical semantics of spatial relations for a language. By providing a language-independent semantic interface to the graphics generation component of WordsEye, the system introduced in this chapter also paves the way for text-to-scene systems in other languages. The ability to generate a 3D scene directly from primitive semantic relations means that a text-to-scene system for a language can be created by mapping text to its underlying semantics and then using the 3D graphics functionality provided by WordsEye to generate a 3D scene. We will use this functionality in Chapter 6, Section 6.5, when we introduce the text-to-scene pipeline for SpatialNet.

Chapter 6

SpatialNet: A Declarative Resource for Spatial Relations

6.1 Introduction

Spatial language understanding is a research area in natural language processing with applications from robotics and navigation to paraphrase and image caption generation. However, most work in this area has been focused specifically on English. While there is a rich literature on the realization of spatial relations in different languages, there is no comprehensive resource which can represent spatial meaning in a formal manner for multiple languages. The development of formal models for the expression of spatial relations in different languages is a largely unstudied but very important problem. In Chapter 1, Section 1.1.2, for example, we saw how machine translation between English and German produced incorrect results due to the differing lexicalization patterns in the two different languages. Another research area that has been very focused on English is text-to-scene generation, most of which continues to use English as the input language. Extending text-to-scene generation to other languages (and potentially other cultural frameworks) is another important problem that should be considered.

In this chapter,[1] we address the issue of modeling cross-linguistic differences in the ex-

[1]Some of the work in this chapter was originally presented in Ulinski *et al.* [2019].

pression of spatial language by developing a deep semantic representation of spatial relations called *SpatialNet*. SpatialNet is based on FrameNet [Baker *et al.*, 1998; Ruppenhofer *et al.*, 2016] and VigNet [Coyne *et al.*, 2011a], two resources which use frame semantics to encode lexical meaning. SpatialNet provides a formal description of the lexical semantics of spatial relations by linking linguistic expressions both to semantic frames and to actual spatial configurations. This formal representation of the lexical semantics of spatial language also provides a consistent way to represent spatial meaning across multiple languages. SpatialNet can be used in conjunction with the graphics generation component of the WordsEye text-to-scene system to produce a 3D scene from a spatial description. This means that creating SpatialNet resources for multiple languages can also facilitate multilingual text-to-scene generation.

SpatialNet is divided into separate modules. *Spatio-graphic primitives* (SGPs) represent possible graphical (spatial) relations. The *ontology* represents physical objects and their classification into semantic categories. Both are based on physical properties of the world and do not depend on a particular language. *Spatial frames* are language-specific (though, like the frames of FrameNet, may be shared among many languages) and represent the lexical meanings a language expresses. *Spatial vignettes* group together lexical items, spatial frames, and SGPs with spatial and graphical constraints from the ontology, grounding the meaning in a language-independent manner.

In this chapter, we describe the structure of SpatialNet, using examples from our pilot work on English and German. We also show how SpatialNet can be combined with other existing natural language processing tools to create a text-to-scene system for a language. In Section 6.2, we discuss related work. In Section 6.3, we review background information on FrameNet and VigNet. In Section 6.4, we describe the structure of SpatialNet, with English and German examples. In Section 6.5, we show how the SpatialNet for a language can be used in conjunction with the WordsEye text-to-scene system to generate 3D scenes from input text in that language. We conclude in Section 6.6 and discuss future work.

6.2 Related Work

Spatial relations have been studied in linguistics for many years. For example, Herskovits [1986] catalogs fine-grained distinctions in the interpretation of various English prepositions, and Feist and Gentner [1998] show that the shape, function, and animacy of the figure and ground objects are factors in the perception of spatial relations as *in* or *on*. Bowerman and Choi [2003] describe how Korean linguistically differentiates between putting something in a loose-fitting container (*nehta*, e.g. fruit in a bag) vs. in a tight fitting wrapper (*kkita*, e.g. hand in glove). Other languages (English included) do not make this distinction. Levinson [2003] and colleagues have also catalogued profound differences in the ways different languages encode relations between objects in the world. Our work differs from linguistic efforts such as these in that our work results in a formal representation of how each language expresses spatial information. This will allow our work to easily be applied to many other problems in NLP. Since the representation is human- as well as machine-readable, it can also be used in more traditional linguistics studies.

Another area of research focuses on computational processing of spatial language. Pustejovsky [2017] has developed an annotation scheme for labeling text with spatial roles. This type of annotation can be used to train classifiers to automatically perform the task, as demonstrated by the SpaceEval task [Pustejovsky *et al.*, 2015]. Although this work provides examples of how a language expresses spatial relations, annotation of spatial roles does not provide a formal description of the link between surface realization and underlying semantics. Our work provides a formal description and also a semantic grounding that tells us the actual spatial configuration denoted by a set of spatial roles. Also, our work extends to languages other than English.

Petruck and Ellsworth [2018] advocate using FrameNet [Ruppenhofer *et al.*, 2016] to represent spatial language. FrameNet uses frame semantics to encode lexical meaning. VigNet [Coyne *et al.*, 2011a] is an extension of FrameNet used in the WordsEye text-to-scene system. SpatialNet builds on conventions used in both FrameNet and VigNet; we have discussed FrameNet and VigNet in previous chapters of this thesis and will review some of this background information in Section 6.3.

6.3 Background on FrameNet and VigNet

In this section, we review some of the relevant background on FrameNet and VigNet, focusing on their application to processing spatial language. FrameNet encodes lexical meaning using a frame-semantic conceptual framework. In FrameNet, lexical items are grouped together in *frames* according to shared semantic structure. Every frame contains a number of *frame elements* (semantic roles) which are participants in this structure. Words that evoke a frame are called *lexical units*. A lexical unit is also linked to sentences that have been manually annotated to identify frame element fillers and their grammatical functions. This results in a set of *valence patterns* that represent possible mappings between syntactic functions and frame elements for the lexical unit. FrameNet already contains a number of frames for spatial language. Spatial language frames in FrameNet inherit from LOCATIVE-RELATION, which defines core frame elements FIGURE and GROUND, as well as non-core frame elements including DISTANCE and DIRECTION. Examples of spatial language frames are SPATIAL-CONTACT, CONTAINMENT and ADJACENCY.

VigNet, a lexical resource inspired by and based on FrameNet, was developed as part of the WordsEye text-to-scene system. VigNet extends FrameNet in several ways. It adds much more fine-grained frames, primarily based on differences in graphical realization. For example, the verb *wash* can be realized in many different ways, depending on whether one is washing dishes or one's hair or a car; VigNet therefore has several different wash frames. VigNet also adds graphical semantics to frames. It does this by adding primitive graphical (typically, spatial) relations between frame element fillers. These graphical relations can represent the position, orientation, size, color, texture, and poses of objects in the scene. The graphical semantics can be thought of as a semantic grounding; it is used by WordsEye to construct and render a 3D scene. Frames augmented with graphical semantics are called *vignettes*.

Information about the 3D objects in WordsEye is organized in VigNet into an *ontology*. The ontology is a hierarchy of semantic types with multiple inheritance. Types include both 3D objects and more general semantic concepts. For example, a particular 3D rocking chair is a sub-type of ROCKING-CHAIR.N. Every 3D object has a semantic type and is inserted into the ontology. WordsEye also includes lexicalized concepts (e.g. *chair* tied to CHAIR.N) in the ontology. The ontology includes a knowledge base of assertions that provide more information about semantic concepts. Assertions include sizes of objects and concepts, their parts, their colors, what they typically contain, what affordances they have, and information about their function. Spatial affordances and other properties can be applied to both 3D graphical objects and to more general semantic types. For example, the general semantic type CUP.N has a CUPPED REGION affordance, since this affordance is shared by all cups. A particular 3D graphical object of a cup might have a HANDLE affordance, while another might have a LID affordance, but these spatial affordances are not tied to the super-type CUP.N.

6.4 Structure of SpatialNet

SpatialNet provides a formal description of spatial semantics by linking linguistic expressions to semantic frames and linking semantic frames to actual spatial configurations. To do this, we adopt some conventions from FrameNet and VigNet, making some changes to address some of the shortcomings of these resources.

FrameNet provides semantic frames including frames for spatial language. However, the syntactic information provided in the valence patterns is often insufficient for the purpose of automatically identifying frame elements in new sentences. One example is frames where the target word is a preposition, which includes many of the frames for spatial language. According to the FrameNet annotation guidelines for these [Ruppenhofer *et al.*, 2016, page 50], the GROUND is assigned the grammatical function Obj(ect), and the FIGURE is tagged as an Ext(ernal) argument. Given a previously unseen sentence, automatic methods can identify the object of the preposition and therefore the GROUND, but the sentence may contain several noun phrases outside the prepositional phrase, making the choice of FIGURE ambiguous. FrameNet also does not provide a semantic grounding. To create SpatialNet, we adopt the concept of a FrameNet *frame*, including the definition of *frame elements* and *lexical units*. However, we modify the convention for the valence patterns to more precisely define syntactic patterns in a declarative format. In addition, to facilitate the use of SpatialNet across different languages, we specify the syntactic constraints in valence patterns using labels from the Universal Dependencies project [Universal Dependencies].

VigNet does provide a grounding in graphical semantics, but presents other problems. First, VigNet does not currently include a mapping from syntax to semantic frames. In addition, although vignettes provide a framework for linking semantic frames to primitive graphical relations, the VigNet resource does not include frames for spatial prepositions, but only for higher-level semantic constructs. Finally, since VigNet has been developed specifically for English, some parts of the existing resource do not generalize easily to other languages. To create SpatialNet, we adopt from VigNet the concept of a *vignette* and the *semantic ontology*. However, we make the resource more applicable across languages by (a) formalizing the set of primitive graphical relations and constraints used in vignettes into what we call *spatio-graphic primitives* (SGPs), and (b) moving the language-specific mapping of lexical items to semantic categories out of the VigNet ontology and into a separate database. The SGPs and semantic ontology are used to define a language-independent semantic grounding for vignettes.

A SpatialNet for a particular language consists of a set of *spatial frames*, which link surface language to lexical semantics using valence patterns, and a set of *spatial vignettes*, which link spatial frames and lexical units to SGPs based on semantic/functional constraints. For our pilot work on SpatialNet, we are developing SpatialNet resources for English and German. In the following sections, we will describe these modules in more detail, using examples from our work on English and German.

6.4.1 Ontology of Semantic Categories

The ontology in VigNet consists of a hierarchy of semantic types (concepts) and a knowledge base containing assertions. SpatialNet uses the VigNet ontology and semantic concepts directly, under the assumption that the semantic types and assertions are language-independent. Thus far, our work on English and German has not required modification of the ontology; however, since it was developed for English, it may need to be extended or modified in the future to be relevant for other languages and cultures. VigNet also includes lexicalized concepts (e.g. *chair* tied to CHAIR.N) in the ontology. For SpatialNet, we store this language-dependent lexical information in a separate database.

The mapping from lexical items to semantic concepts is important for the decomposition of text into semantics. For English SpatialNet, we use the lexical mapping extracted from VigNet. To facilitate creation of lexical mappings for other languages, we mapped VigNet concepts to entries in the Princeton WordNet of English [Princeton University]. A mapping was constructed as follows: For each lexicalized concept in VigNet, we looked up each of its linked lexical items in WordNet. If the word (with correct part of speech) was found in WordNet, we added mappings between the VigNet concept and each WordNet synset for that word. This resulted in a many-to-many mapping of VigNet concepts to WordNet synsets.

To obtain a lexical mapping for German, we use the VigNet–WordNet map in conjunction with GermaNet [Henrich and Hinrichs, 2010; Hamp and Feldweg, 1997]. GermaNet includes mappings to Princeton WordNet 3.0. For a given German lexical item, we use the GermaNet links to Princeton WordNet to obtain a set of possible VigNet concepts from the VigNet–WordNet mapping. We are also experimenting with the Open German WordNet [Siegel], although in general we have found it to be less accurate. Open German WordNet includes links to the EuroWordNet Interlingual Index (ILI) [Vossen, 1998], which are in turn mapped to the Princeton English WordNet. Table 6.1 shows the VigNet concepts for some German words, obtained using GermaNet and Open German WordNet.

6.4.2 Spatio-graphic Primitives

To create the set of spatio-graphic primitives used in SpatialNet, we began with primitive spatial and graphical relations already in VigNet. These were introduced in Chapter 5, Section 5.3.2, and include about 100 primitive relations. We wanted to make sure that this set of primitives was as comprehensive as possible, and not limited to the current graphical capabilities of WordsEye. To that end, we annotated the 71 pictures in the Topological Relations Picture Series [Bowerman and Pederson, 1992] and the 68 pictures in the Picture Series for Positional Verbs [Ameka *et al.*, 1999] with the spatial and graphical primitives

Lexical item	**VigNet concepts**	
	GermaNet	**ODE-WordNet**
Mauer	WALL.N	WALL.N
	RAMPART-WALL.N	
	RAMPART.N	
Katze	DOMESTIC-CAT.N	DOMESTIC-CAT.N
	HOUSE-CAT.N	HOUSE-CAT.N
		TRUE-CAT.N
Gemälde	PAINTING.N	PICTURE.N
	PICTURE.N	ICON.N
		IMAGE.N
Haus	HOUSE.N	SHACK.N
		HUTCH.N
		HOUSE.N
		FAMILY.N
		HOME.N

Table 6.1: Mapping from German lexical items to VigNet semantic categories, obtained using two different German WordNet resources: GermaNet and Open German WordNet.

represented by each image. When an appropriate spatial primitive did not exist in VigNet, we created a new one. These new primitives have also been added to a list of "pending" graphical relations in VigNet that the WordsEye developers plan to implement in the future. Some examples of primitives that were added to VigNet are shown in Figure 6.1. In total, we added about 70 new primitives, for a total of 175 SGPs. The complete set of SGPs is provided in Appendix D.

We use WordsEye as a realization engine for the SGPs. This is done using the modified WordsEye architecture and web API introduced in Chapter 5, Section 5.3. The semantic

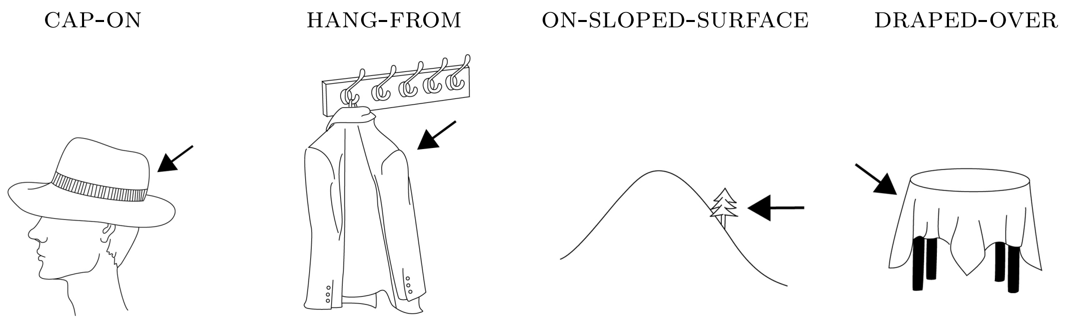


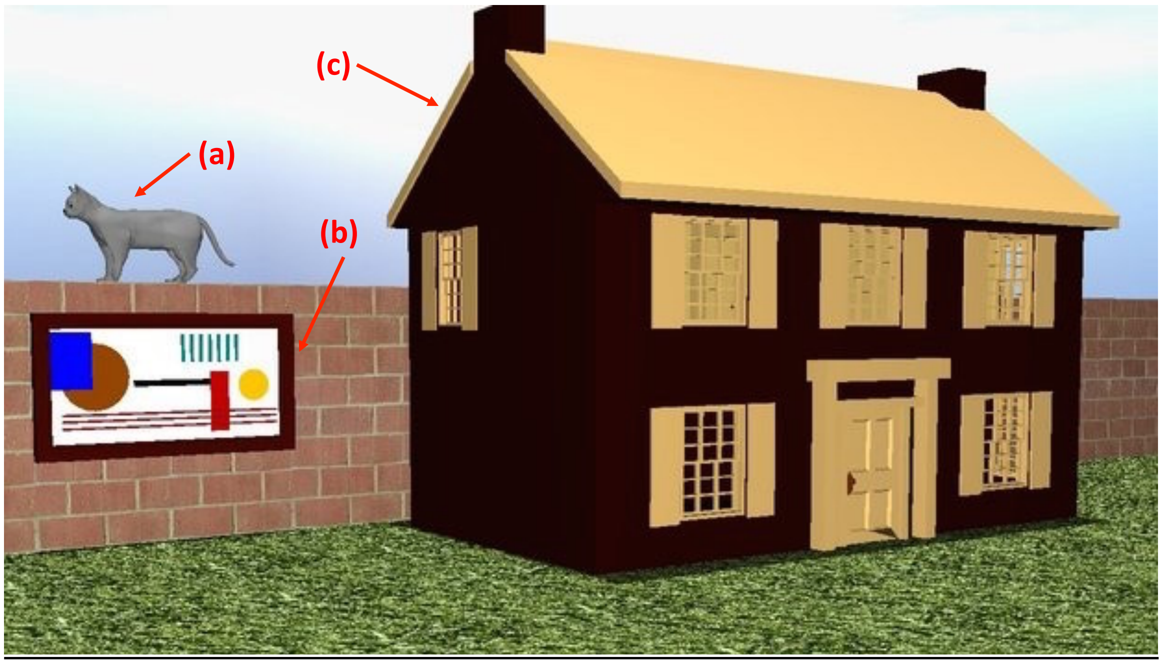
Figure 6.1: Examples of new spatial primitives added for SpatialNet

representation consists of a list of entities, each with a semantic type from the VigNet ontology, and a list of relations between entities. SpatialNet SGPs can be used as relations in this semantic input; we are working closely with the WordsEye developers to ensure that SGPs in SpatialNet continue to be compatible with the WordsEye system. Relations that are marked as "pending" or are otherwise unsupported by the graphics component of WordsEye are ignored and not included in the 3D graphics. The entities referenced by these relations will be displayed in a default position (side-by-side). Figure 6.2 shows a scene created in WordsEye that demonstrates the spatio-graphic primitives ON-TOP-SURFACE, ON-FRONT-SURFACE, and NEXT-TO.

Note that not all SGPs will be applicable to all languages. For example, the Australian language Guugu Yimithirr uses only absolute frames of reference to refer to the positions of objects [Levinson, 2003]. A SpatialNet created for such a language will not include SGPs like LEFT-OF and IN-FRONT-OF, which are based on relative and intrinsic frames of reference.

6.4.3 Spatial Frames

Spatial frames represent the lexical meanings a language can express. The structure of spatial frames is closely based on FrameNet frames. We have incorporated many of the FrameNet spatial language frames into SpatialNet, adding to these as needed. For example, for English we have added an ON-SURFACE frame that inherits from SPATIAL-CONTACT. The main difference between SpatialNet frames and FrameNet frames is in the definition



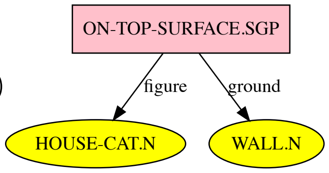
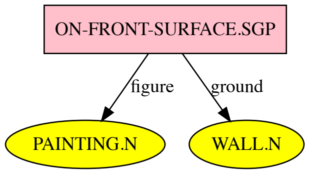
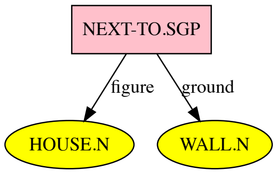
	SGP	English	German
(a)		*The cat is* ***on*** *the wall.*	*Die Katze ist* ***auf*** *der Mauer.*
(b)		*The painting is* ***on*** *the wall.*	*Das Gemälde ist* ***an*** *der Mauer.*
(c)		*The house is* ***at*** *the wall.*	*Das Haus ist* ***an*** *der Mauer.*

Figure 6.2: Examples of spatio-graphic primitives: (a) ON-TOP-SURFACE, (b) ON-FRONT-SURFACE, and (c) NEXT-TO and English/German descriptions.

```
<frame name="On_surface">
  <parent name="Spatial_contact"/>
  <FE name="Figure"/>
  <FE name="Ground"/>
  <lexUnit name="on_top_of.adp">
    <pattern>
      <dep FE="Ground" tag="NOUN">
        <dep FE="Figure" reln="nsubj"/>
        <dep reln="case" word="on">
          <dep word="top" reln="mwe"/>
          <dep word="of" reln="mwe"/>
        </dep>
      </dep>
    </pattern>
  </lexUnit>
  <lexUnit name="on.adp">
    <pattern>
      <dep FE="Ground" tag="NOUN">
        <dep FE="Figure" reln="nsubj"/>
        <dep reln="case" word="on"/>
      </dep>
    </pattern>
  </lexUnit>
</frame>
```

Figure 6.3: Declarative format for spatial frames

of the valence patterns. SpatialNet defines valence patterns by precisely specifying lexical and syntactic constraints, which can be based on the syntactic dependency tree structure, grammatical relations, parts of speech, or lexical items. Figure 6.5, which provides examples of spatial vignettes for English, includes a valence pattern for the English lexical unit *on.adp*. This pattern specifies a syntactic structure consisting of a root (which must have part of speech `NOUN`), an `nsubj` dependent, and a `case` dependent (which must be the word *on*). The declarative format used to define this spatial frame is shown in Figure 6.3.

```
<vignette name="on-vertical-surface">
  <input frame="On_surface" lexUnit="on.adp"/>
  <type-constraint FE="Ground" type="vertical-surface.n"/>
  <type-constraint FE="Figure" type="wall-item.n"/>
  <output relation="on-front-surface.r">
    <map FE="Ground" arg="ground"/>
    <map FE="Figure" arg="figure"/>
  </output>
</vignette>
<vignette name="on-top-surface">
  <input frame="On_surface" lexUnit="on.adp"/>
  <input frame="On_surface" lexUnit="on_top_of.adp"/>
  <type-constraint FE="Ground" type="upward-surface.n"/>
  <output relation="on-top-surface.r">
    <map FE="Ground" arg="ground"/>
    <map FE="Figure" arg="figure"/>
  </output>
</vignette>
```

Figure 6.4: Declarative format for spatial vignettes

6.4.4 Spatial Vignettes

Spatial vignettes use spatial frames, SGPs, and the ontology to interpret prepositions and other lexical information in a language. They relate linguistic realization (e.g. a preposition with its argument structure) to a spatial frame (such as ON-SURFACE), and at the same time to a graphical semantics expressed in terms of SGPs and additional constraints. This lexical information is often ambiguous, as demonstrated by the English and German descriptions in Figure 6.2. In English, the preposition *on* is ambiguous; it can mean either ON-TOP-SURFACE or ON-FRONT-SURFACE. In German, the preposition *an* is ambiguous; it can mean either ON-FRONT-SURFACE or NEXT-TO. To resolve such ambiguities, vignettes place selectional restrictions on frame elements that require fillers to have particular spatial affordances, spatial properties (such as the object size, shape, and orientation), or functional properties (such as whether the object is a vehicle or path). This information is found in the ontology.

Consider the spatial vignettes that would be used to disambiguate the meanings of English *on* in the Figure 6.2 examples. The declarative format used to define these spatial

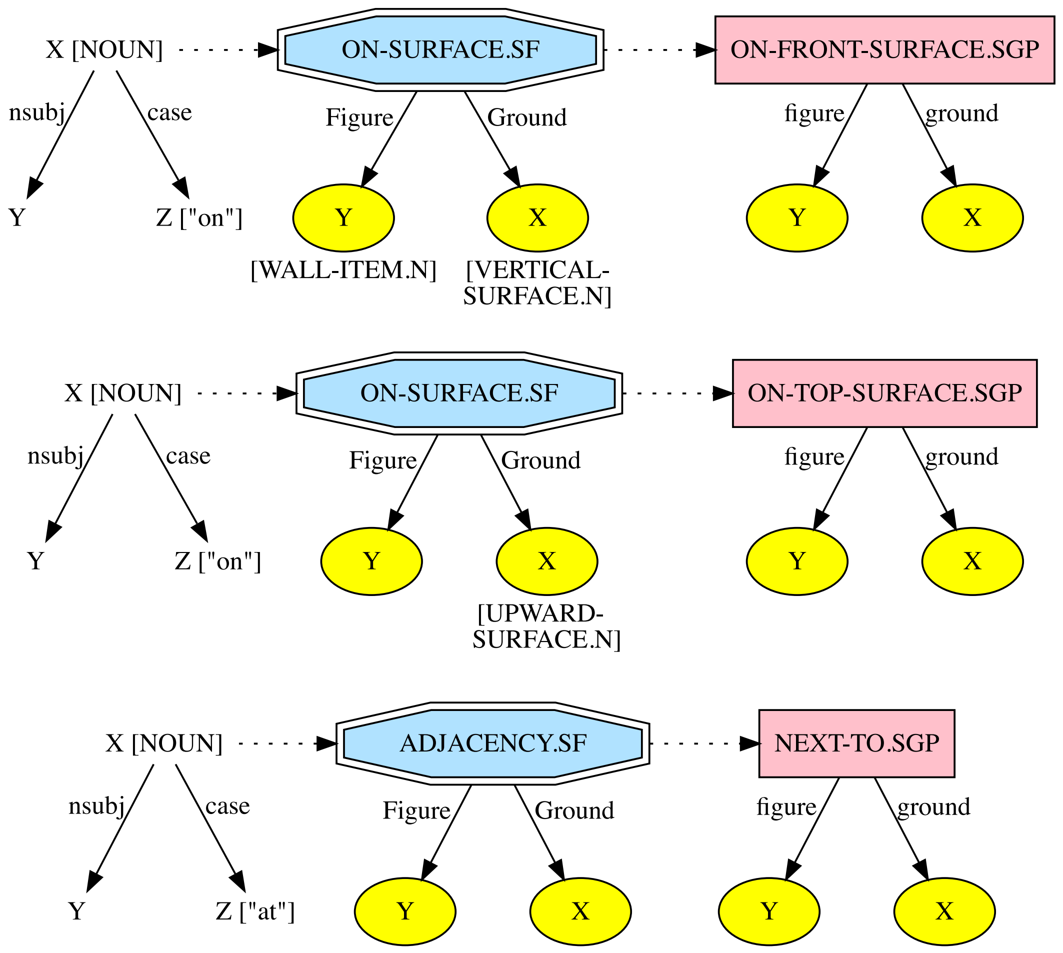


Figure 6.5: Spatial vignettes for different meanings of English prepositions. Vignettes resolve the spatial relation given the spatial and functional object features. Spatial frames are represented by blue octagons, and SGPs by pink rectangles.

vignettes is shown in Figure 6.4. A visual representation of the vignettes is shown in Figure 6.5. The first two vignettes link the ON-SURFACE spatial frame to different SGPs based on features of the frame element fillers. The first vignette, which links the preposition *on* from the ON-SURFACE spatial frame to the ON-FRONT-SURFACE SGP, adds semantic type constraints to both the Figure and the Ground. The Figure must be of type WALL-ITEM.N and the Ground must be of type VERTICAL-SURFACE.N. If these constraints are met, the vignette produces the SGP ON-FRONT-SURFACE as output, mapping Figure to the SGP argument

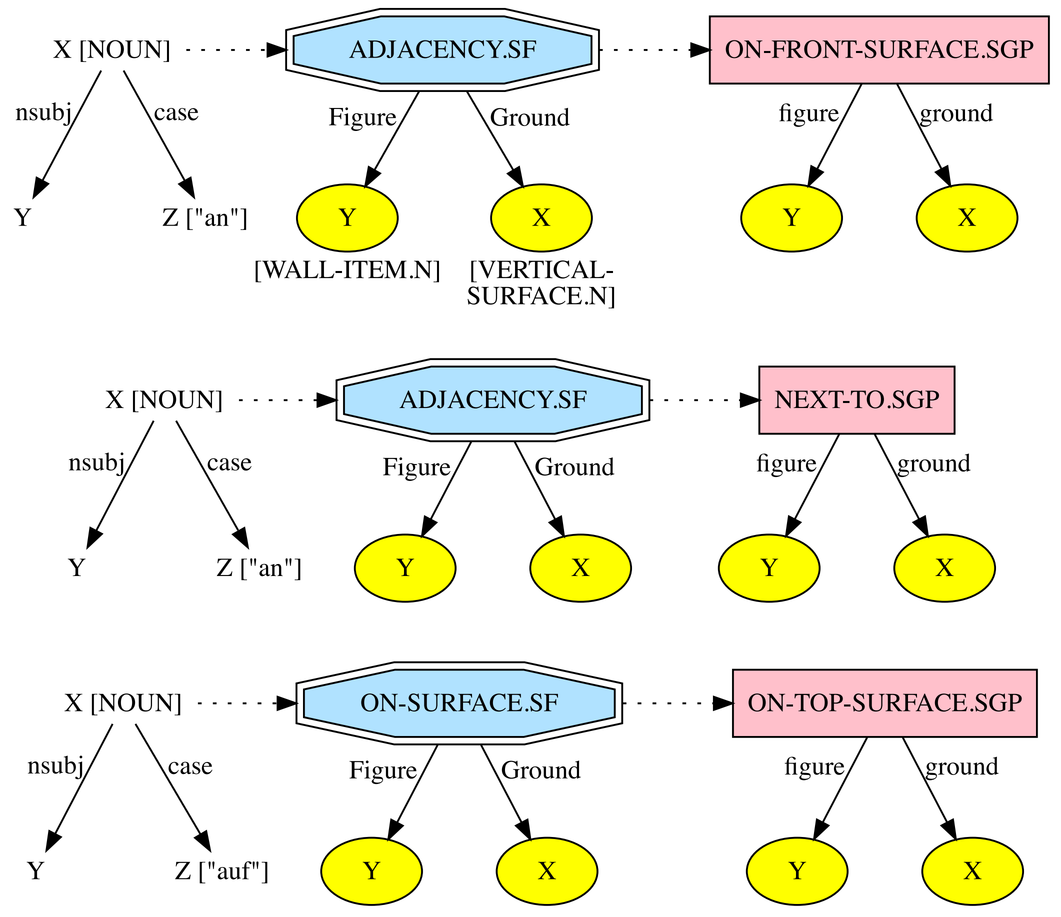


Figure 6.6: Spatial vignettes for different meanings of German prepositions. Vignettes resolve the spatial relation given the spatial and functional object features. Spatial frames are represented by blue octagons, and SGPs by pink rectangles.

`figure`, and `Ground` to the SGP argument `ground`. The second vignette, which links *on* to the ON-TOP-SURFACE SGP, has a semantic type constraint only on the `Ground`, requiring it to be of type UPWARD-SURFACE.N. If this constraint is met, the vignette produces the SGP ON-TOP-SURFACE. Note that while in this case the frame elements and SGP arguments have the same names, this is not necessarily true for all vignettes (see, for example, the WordsEye vignettes in Figure 2.3). Note also that in English, *painting **on** wall* is actually ambiguous, since a painting can balance on the top of a wall as well as hang on its front surface (although the latter interpretation is perhaps more likely). The spatial vignettes

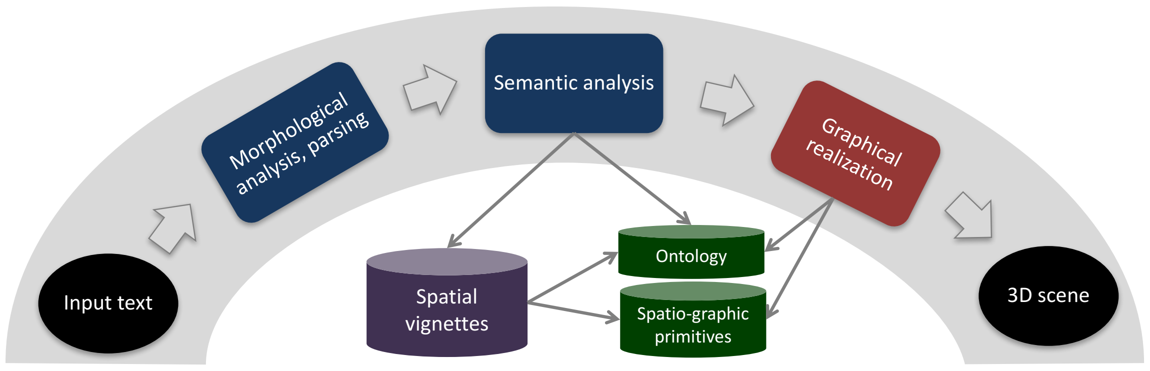
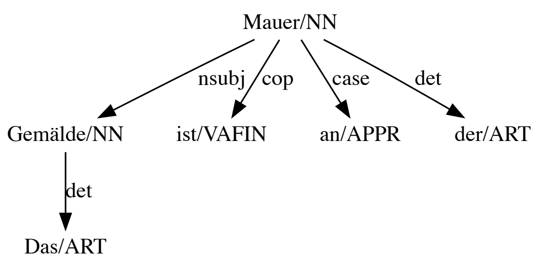


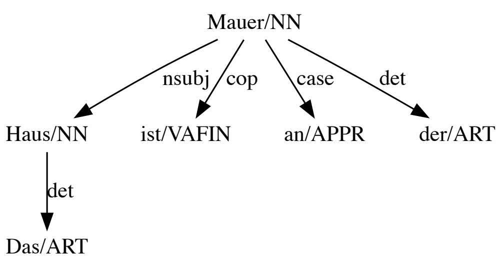
Figure 6.7: Pipeline for text-to-scene generation with SpatialNet

allow for either interpretation. The third English vignette in Figure 6.5 interprets the preposition *at* by linking it to the ADJACENCY spatial frame and the NEXT-TO SGP with no semantic type constraints.

Figure 6.6 shows the vignettes which would be used to disambiguate the meanings of German *an* from the sentences in Figure 6.2. The first two German vignettes link the ADJACENCY spatial frame to SGPs. The first vignette, which links the preposition *an* from the ADJACENCY spatial frame to the SGP ON-FRONT-SURFACE, is identical to the first English vignette in Figure 6.5, except for the input frame and lexical unit. The semantic type constraints, SGPs, and frame element to SGP argument mappings are the same. The second vignette, which links *an* to the NEXT-TO SGP, does not have any semantic type constraints. It outputs the SGP NEXT-TO, mapping Figure to the SGP argument `figure` and Ground to the SGP argument `ground`. A third German vignette interprets the preposition *auf*, linking it to the ON-SURFACE spatial frame and the ON-TOP-SURFACE SGP. In the next section, we will walk through a complete example of using spatial vignettes to interpret German sentences, in the context of text-to-scene generation.



(b)



(c)

Figure 6.8: Results of morphological and syntactic analysis of German sentences (b) *Das Gemälde ist* ***an*** *der Mauer* and (c) *Das Haus ist* ***an*** *der Mauer*.

6.5 Using SpatialNet for Text-to-Scene Generation

Although in many areas of natural language processing there is an increasing emphasis on multilinguality and cross-lingual resources, most text-to-scene research continues to use English as the input language. There is some work on text-to-scene systems for other languages, e.g. Turkish [Kılıçaslan *et al.*, 2008], Russian [Ustalov, 2012], and Hindi [Jain *et al.*, 2017]). However, these efforts are isolated and do not make substantial use of prior research in English text-to-scene generation. SpatialNet can be used in conjunction with the graphics generation component of the WordsEye text-to-scene system to produce a 3D scene from a spatial description. This means that creating a SpatialNet resource for a language also facilitates text-to-scene generation for that language. Figure 6.7 shows an overview of our system for text-to-scene generation. Although SpatialNet focuses on semantics, the system also requires modules for morphological analysis and syntactic parsing. For English and German, we use the Stanford CoreNLP Toolkit [Manning *et al.*, 2014]. In this section, we describe how we use Stanford CoreNLP, SpatialNet, and WordsEye to convert text into a 3D scene. We illustrate using German sentences (b) and (c) from Figure 6.2.

First, Stanford CoreNLP is used to perform lemmatization, part-of-speech tagging, and dependency parsing. Figure 6.8 shows the resulting dependency structures. The dependency structures are matched against the valence patterns in spatial frames. Sentences (b) and (c) both match the valence pattern for the lexical unit *an.adp* in the ADJACENCY frame.

The valence pattern identifies which lexical items in the sentence will act as frame element fillers. These lexical items are converted into semantic concepts using the lexical mapping from Section 6.4.1. We refer to Table 6.1 to obtain the semantic concepts for the German lexical items. For the purposes of this example, we select the first semantic concept from the GermaNet mapping, which maps *Gemälde* to PAINTING.N, *Mauer* to WALL.N, and *Haus* to HOUSE.N.

The system then identifies the spatial vignettes which accept the frame and lexical unit as input. The features of the semantic concepts obtained for each frame element are checked against the semantic constraints in these spatial vignettes. For German sentence (b), since a WALL.N is a VERTICAL-SURFACE.N and a PAINTING.N is a WALL-ITEM.N, the vignette which decomposes into ON-FRONT-SURFACE is a possible match. Since a WALL.N is also an UPWARD-SURFACE.N, the vignette which decomposes into ON-TOP-SURFACE is also a possible match. For now, we select the first matching vignette, which produces the SGP ON-FRONT-SURFACE with `figure`=PAINTING.N and `ground`=WALL.N. For German sentence (c), since HOUSE.N is not a WALL-ITEM.N, only the vignette which decomposes into NEXT-TO is matched. This produces the SGP NEXT-TO, with `figure`=HOUSE.N and `ground`=WALL.N. The entities and SGPs for each sentence are then converted into a semantic representation compatible with the modified WordsEye web API (shown in Figure 6.9), which is used to generate a 3D scene.

6.6 Conclusion

We have described our development of a novel resource, SpatialNet, which provides a formal representation of how a language expresses spatial relations. We have discussed the structure of the resource, including examples from our pilot work developing English and German SpatialNet resources. We have also introduced a text-to-scene generation pipeline for using SpatialNet to convert text into 3D scenes; this pipeline can support text-to-scene generation for any language that has both a syntactic dependency parser and a SpatialNet resource.

The first area for future work on SpatialNet is to incorporate it into the WELT L2 tools for language documentation, which we introduced in Chapter 3. The WELT documentation

```
{"entities": [
    {"id":"n.1",
        "isa":["painting.n"]
    },
    {"id":"n.2",
        "isa":["wall.n"]
    }
 ],
 "relations": [
    {"id":"r.1",
     "isa":"gfx.on-front-surface.r",
     "core-args": {
        "figure":"n.1",
        "ground":"n.2"
     }
   }
 ]
}
```

(b)

```
{"entities": [
    {"id":"n.1",
        "isa":["house.n"]
    },
    {"id":"n.2",
        "isa":["wall.n"]
    }
 ],
 "relations": [
    {"id":"r.1",
     "isa":"gfx.next-to.r",
     "core-args": {
        "figure":"n.1",
        "ground":"n.2"
     }
   }
 ]
}
```

(c)

Figure 6.9: Final semantic representation for German sentences (b) *Das Gemälde ist an der Mauer* and (c) *Das Haus ist an der Mauer*, used to generate a 3D scene with WordsEye.

tools allow the formal documentation of the semantics of a language by specifying syntax-to-semantics rules. These can be incorporated into the text-to-scene generation pipeline of the WordsEye desktop application, thereby enabling the generation of 3D scenes from input text in the endangered language, which can be used to verify the accuracy of the semantic documentation with endangered language speakers. Currently, the syntax-to-semantics rules created in WELT L2 are mapped into VigNet, which is (a) specific to the English language and (b) often biased toward Western culture. The solution we proposed for this in Chapter 3 was that the vignettes in WordsEye could be edited and/or created to fit the endangered language. We showed how a vignette could be modified for this purpose in Section 3.7.2, by overriding the default values for its arguments. However, this method relies on the fact that an appropriate vignette already exists in VigNet for us to modify. Defining new vignettes, or even adding new arguments to existing vignettes, requires WordsEye developers to manually edit the WordsEye system and provide a patch for the desktop program. Using

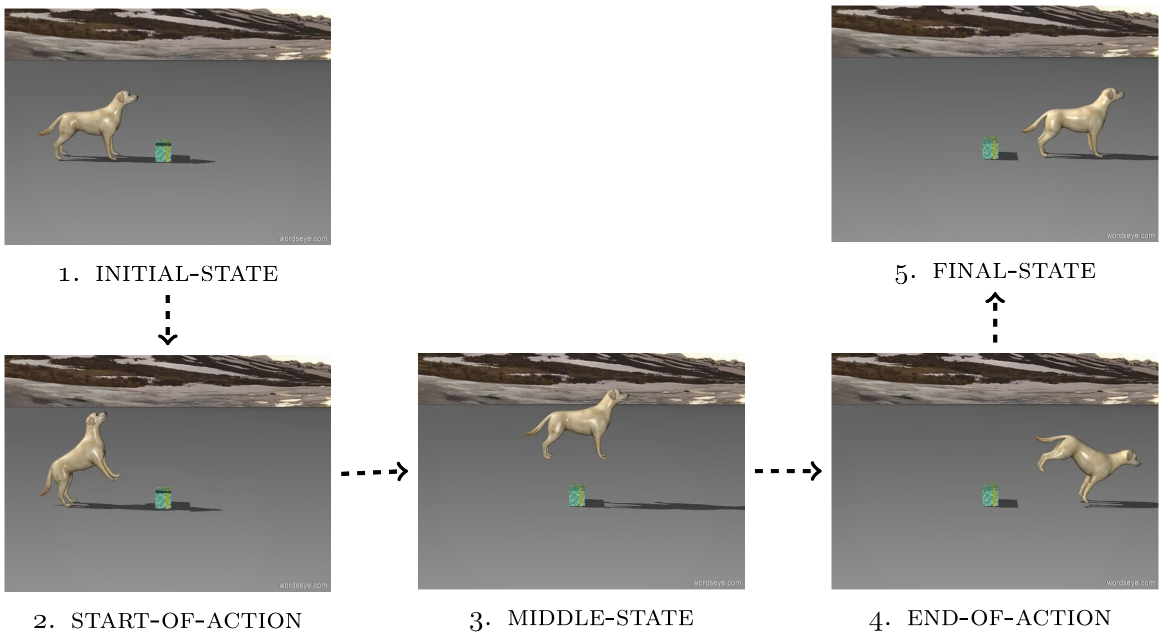


Figure 6.10: A possible sequence of static scenes to represent a dog jumping over a box

SpatialNet rather than VigNet for the semantic documentation in WELT L2 would allow us to create syntax-to-semantics rules without direct assistance from WordsEye developers. In addition, using SpatialNet would ensure that our syntax-to-semantics mapping is to a language-independent semantics, removing any bias toward English lexical semantics in the semantic documentation produced by WELT L2. The example vignettes provided in this chapter are used to interpret basic spatial prepositions, which can in general be mapped into a single spatio-graphic primitive. SpatialNet also supports including multiple SGPs in the output for a vignette, which means that it can be used for more concepts requiring more complex graphical semantics, such as *kicking a goal*.

A second area for future work on SpatialNet is to extend the semantic representation to handle motion as well as static spatial relations. Like VigNet, SpatialNet could be used to provide graphical semantics for motion verbs by representing them as a static slice of time; for example, defining *walk toward* to include a figure in a *walking* pose, midway between the source and the goal, and facing toward the goal. To more precisely define the lexical semantics of motion verbs, we can instead introduce the concept of a *motion vignette*, which is represented by a labeled sequence of static sets of SGPs associated with key stages of that action. These stages might include INITIAL-STATE, START-OF-ACTION, MIDDLE-STATE, END-OF-ACTION, FINAL-STATE. For example, *The dog jumped off the log* could be represented by the dog standing on the log, the dog leaping off with legs still on the log, the dog in mid air, the front paws touching the ground, and the dog on the ground. Figure 6.10 shows a possible sequence of stages for a dog jumping over a box.

Another research area to explore would be to use SpatialNet to improve machine translation, helping to correct the kinds of errors described in Chapter 1, Section 1.1.2. Both Google Translate and Bing Translator have difficulty translating spatial prepositions between English and German because of the different ways these languages map lexical information into spatial relations. SpatialNet could be used to assist with machine translation of spatial prepositions, case markers with spatial meaning, and even motion predicates if SpatialNet is extended to handle motion. This could be done by re-ranking or post-processing the output of machine translation systems, or by using SpatialNet to provide additional features within the MT system itself.

Finally, a further avenue for future work will be to explore ways of automatically (or semi-automatically) populating SpatialNet, which would facilitate the process of extending SpatialNet to other languages. We will discuss this possibility in more detail in Chapter 7, Section 7.3.1.

Chapter 7

Language Elicitation via Crowdsourcing

7.1 Introduction

In the earlier chapters of this thesis, we have shown how 3D scenes generated with WordsEye can be used by a field linguist to elicit spatial language from an endangered language speaker. In this chapter, we demonstrate three other ways we have used WordsEye to facilitate the elicitation of language, all of which utilize crowdsourcing.

3D scenes are useful not only for the elicitation of small amounts of data in a field linguistics setting, but also for the elicitation larger amounts of data for natural language processing applications. In our first case study, which we will describe in Section 7.2, we use 3D scenes previously created by WordsEye users and published to the WordsEye gallery to collect a corpus of text that we use to train machine learning classifiers to automatically detect emotion from textual descriptions. Next, in Section 7.3, we elicit spatial descriptions from 3D scenes generated using the system introduced in Chapter 5, as part of a pilot study to explore the possibility of an active learning system for SpatialNet. Finally, in Section 7.4, we use the underlying semantic model contained in VigNet to elicit depictive sentences that are not restricted by the content of pre-existing images, resulting in a set of *imaginative sentences.*

7.2 Elicitation of Emotional Language

In this section,[1] we describe how we use 3D scenes created in WordsEye to elicit descriptions containing emotional language. Others have used crowdsourcing to elicit human-generated sentences, e.g. to create image captions. This includes the PASCAL image caption corpus [Rashtchian *et al.*, 2010], Flickr8k [Hodosh *et al.*, 2013] and Microsoft COCO [Chen *et al.*, 2015]. Our work differs in our use of human-created 3D scenes as opposed to corpora of realistic photographs and in that we collect an additional description designed to contain emotional vocabulary.

One function that has not yet been explored for text-to-scene generation, in WordsEye or otherwise, is to set the mood of the scene automatically based on the input text. For example, the system could manipulate the lighting or the predominant colors in the scene to emphasize a particular mood. In addition to the input text used to generate the scene, the WordsEye website allows users to attach titles and captions to the scenes they create. Users can publish their scenes to a public gallery, where other users can subsequently add their own comments on the scene, expressing their opinions and reactions. Our goal for the dataset described in this section is to simulate the kind of text generated by users in the WordsEye web application and gallery. This will help us to learn to automatically associate a mood with a 3D scene. Specifically, we are interested in the Ekman Big Six emotions (happiness, sadness, anger, surprise, fear, and disgust) [Ekman, 1999].

In Section 7.2.1, we describe how we use crowdsourcing to create our corpus. In Section 7.2.2, we describe how we use this corpus to perform machine learning classification experiments to predict emotion from text.

7.2.1 Corpus Collection

In order to create our dataset, we began by compiling a subset of the 3D scenes previously published by users to the WordsEye gallery. In this way, we collected a total of 660 images.

[1]Some of the material in this section was previously published in Ulinski *et al.* [2012], partially supported by by the National Science Foundation under Grant No. 1145505: "EAGER: Using Social Media and Crowdsourcing to Create a New Affect Dictionary."

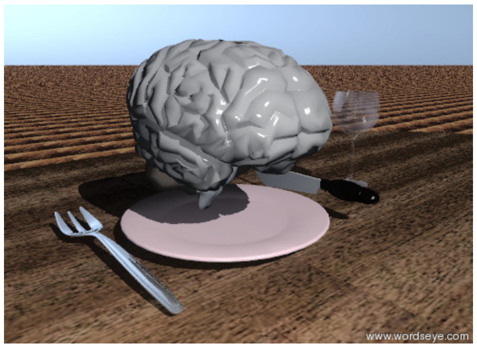
Class	**Total # labels**	**# pictures with 2/3 agreement**	**# pictures with 3/3 agreement**
Happiness	731 (36.9%)	134 (20.3%)	88 (13.3%)
Sadness	267 (13.5%)	50 (7.6%)	8 (1.2%)
Anger	69 (3.5%)	4 (0.6%)	3 (0.5%)
Surprise	430 (21.7%)	88 (13.3%)	13 (2.0%)
Fear	382 (19.3%)	56 (8.5%)	30 (4.5%)
Disgust	101 (5.1%)	14 (2.1%)	3 (0.5%)
Total	**1980 (100%)**	**346 (52.4%)**	**145 (22.0%)**

Table 7.1: Distribution of mood labels in crowdsourced emotion dataset

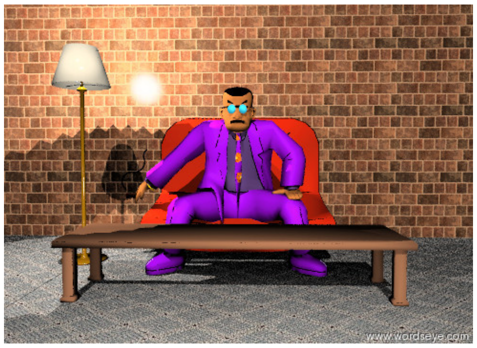
We then created an Amazon Mechanical Turk HIT to obtain for each picture (a) a literal description that could function as a caption for the image, (b) the most relevant mood for the picture, and (c) a short explanation of why the worker selected that mood. For the literal description, we asked workers to write 3-5 sentences (a minimum of 200 characters), avoiding the use of emotional vocabulary. For the mood label, workers were given six choices, corresponding to the Ekman Big Six emotions. For the explanation of their choice of mood, we asked for a minimum of 60 characters answering the question "Why did you choose that mood?" This time, we allowed workers to include emotional vocabulary. In order to encourage the use of emotional language in the explanation and more literal language in the high-level description, we also required that the content of the two answers be significantly different from each other. For each completed task, the worker was rewarded $0.30. Figure 7.1 contains some examples of completed HITs.

We published a total of three HITs for each picture, resulting in each scene being assigned three different captions, comments and mood tags. The full dataset, including WordsEye images, captions, mood labels, and comments, will be available through the Columbia University Academic Commons. Table 7.1 shows the distribution of mood labels in the dataset. Approximately 74.4% of the pictures had a 2/3 or 3/3 majority agreement for the mood.

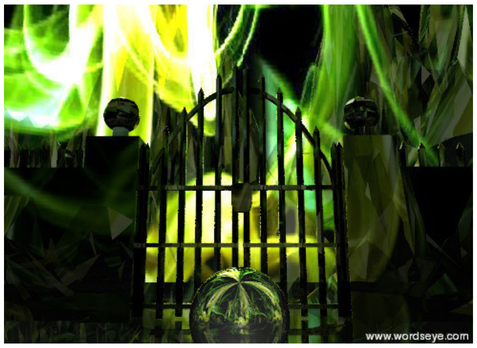
Note that while the corpus we describe in this section was created using WordsEye scenes



Please write a literal description (3-5 sentences) that could work as a caption for the previous image. Avoid the use of emotional vocabulary (min 200 characters):
Right in the middle of a half sown wheat field is a human brain on a plate. The wheat is perfectly lined up on a perfect blue sky day. It is being prepared to be served up for dinner with an elegant glass of wine to accompany it.
Please choose the mood that relates the most to the displayed picture:
Disgust
Why did you choose that mood? Write a short answer (min 60 characters). This time emotional vocabulary is allowed. Make sure the answer is significantly different from the description, or the HIT will not be approved.
Serving up a brain on a platter makes me think of the effects of moving away from hard work and towards tv watching and video game playing has had on us.



Please write a literal description (3-5 sentences) that could work as a caption for the previous image. Avoid the use of emotional vocabulary (min 200 characters):
This is a picture of a man in a purple suit. He is wearing blue sunglasses and is sitting on an orange couch. There is a lamp and a table in the room and the wall is made of brick. There is also a grey carpet in the room.
Please choose the mood that relates the most to the displayed picture:
Anger
Why did you choose that mood? Write a short answer (min 60 characters). This time emotional vocabulary is allowed. Make sure the answer is significantly different from the description, or the HIT will not be approved.
The man in the picture looks angry. His eyebrows make him look that way, due to the way they are slanted.



Please write a literal description (3-5 sentences) that could work as a caption for the previous image. Avoid the use of emotional vocabulary (min 200 characters):
Green, yellow and white lights surround a large black gate, an entrance to something dark and intriguing. There is a figure in front of the door that appears to be a child guarding entrance dearly.
Please choose the mood that relates the most to the displayed picture:
Fear
Why did you choose that mood? Write a short answer (min 60 characters). This time emotional vocabulary is allowed. Make sure the answer is significantly different from the description, or the HIT will not be approved.
This pictures almost looks like the gates into hell and the figure in front of the gate is intense.

Figure 7.1: Completed emotion description HITs

and with WordsEye as our target application, it could be used for automatic mood detection in other contexts as well. For example, Flickr images are likewise tagged with titles, captions, and user comments. An image sharing website like Flickr could, for example, automatically adjust the colors of the user interface surrounding an uploaded picture according to the mood evoked by this text; alternatively, mood-appropriate visual effects could be applied to the pictures themselves, such as adding color overlays or modifying the saturation levels. If the descriptions and reactions to the photographs on such an image sharing website are too different from those found in WordsEye for our corpus to be used directly, the same crowdsourcing methodology could be used to create an emotional language corpus using more relevant images to elicit the descriptions.

7.2.2 Emotion Classification Experiments

In this section, we describe how we use the emotional language corpus we collected to train machine learning classifiers to predict emotion from text. To do this, we created for each image in our corpus a document containing both the caption and the comments. Every reference to the mood in the comments was removed and substituted by the tag <mood> to avoid introducing a bias with the class labels. It would be possible to hone down the dataset by keeping only those images that had a clear majority vote for the mood; that is, images for which 2/3 or 3/3 workers chose identical moods. However, we are interested in subjective and personal opinions, and thus preferred to keep all individual judgments. Since our classification is based only on the descriptive text and not on the source image, it is acceptable to consider each description as a separate datapoint.

7.2.2.1 Methodology

For each example in the dataset a feature vector is built containing the following information:

1. *The class label.*

2. *SentiWordNet scores* [Esuli and Sebastiani, 2006; Baccianella *et al.*, 2010]. SentiWordNet is a lexical tool for opinion mining. It assigns three scores to every WordNet [Miller, 1995; Fellbaum and Miller, 1998] synset: a positivity score, a negativity

score and an objectivity score. The SentiWordNet scores of a document are computed via the mean scores of every word in the document.

3. *LIWC scores* [Tausczik and Pennebaker, 2010]. LIWC is a text analysis tool that calculates 81 language features in several categories including general descriptors, linguistic dimensions, psychological constructs, personal concerns, paralinguistic dimensions and punctuation. For instance, to compute the **posemo** feature, LIWC uses a list of 406 terms that includes words like *love*, *nice*, and *sweet*, and to compute **sad** a list of 101 words is used which comprises terms like *crying* and *grief*.

4. *Dictionary of Affect scores* [Whissell, 1989]. The Dictionary of Affect is a lexical tool to measure the emotional meaning of texts. The DAL assigns activation, evaluation and imagery scores to every word. It does so by comparing each word to a list of 8700+ words rated by their activation, evaluation and imagery. The DAL score of document is given by the average values of its word scores.

5. *TF-IDF of word-POS tag pair*. For each document in the corpus, the TF-IDF score of each possible pair formed by a n-gram word stem and its n-gram part-of-speech tag is computed, with $n = \{1, 2, 3\}$. Our dataset contains approximately 3500 unigrams, 13000 bigrams, and 20000 trigrams.

Starting from the full dataset, 10 balanced datasets are created by sampling the original set without replacement. The prediction estimates over each of these balanced datasets are averaged to eliminate possible bias derived from the random selection of instances. We performed 10-fold cross validation over each balanced dataset. For each fold, we applied feature selection techniques to the training set and, using the selected features, reduced the training and test set to the feature vector lengths: 2, 4, 8, 16, 32, 64, 96, 128, 140, 192, 224, 256, 384 and 500. For each vector length, a search over the SVM cost parameter C was performed using 5-fold cross-validation. The C value that maximized the accuracy of the model was chosen and used to train and test the SVM.

Given the high dimensionality of our prediction problem, it is necessary to apply feature selection techniques to build a smaller and more efficient model. We use the minimum redundancy maximum relevance (mRMR) technique presented in Peng *et al.* [2005], which is found to be effective at selecting relevant features and discarding features redundant with the ones already selected. We executed this technique over each training fold of each balanced dataset to obtain the best 500 features for the prediction problem. The top twenty features are shown in Table 7.2. LIWC and SentiWordNet (SWN) features are shown in bold font; the rest are TF-IDF rates. The ranking was computed by the average min-redundancy max-relevance score of each feature over each partition and fold. The columns contain the rankings for (a) the dataset using exclusively the captions, discarding the comments, and (b) the full dataset, including both captions and comments.

For the full (caption and comment) dataset, most of the best selected features are either TF-IDF scores or LIWC features. This suggests that the sentiment scores computed with SentiWordNet (whose features, except positive score, appear past the ninetieth position) and DAL (whose features appear after the four-hundredth position) are not very useful for our classification problem. It would be interesting to explore the use of more detailed emotion dictionaries that provide scores for individual emotions rather than just polarity, as these would likely be more useful for our task. Most of the features selected by the mRMR technique seem to be consistent with the task at hand, referring to words and concepts intuitively associated with emotions.

7.2.2.2 Experiments and Results

We used the library LIBLINEAR [Fan *et al.*, 2008] for all the classification experiments described in this section. We tried binary classification of each emotion, but the results were not very promising. Instead, we used LIBLINEAR to perform multiway classification. The precision, recall and F-score results of the 6-way SVM classification using the features returned by the mRMR method are shown in Figure 7.2. The accuracy plots are shown in Figure 7.3. Because we balanced the datasets, the baseline accuracy is $1/6 = 16.7\%$. Table 7.3 shows the results obtained using the number of features that maximized the average accuracy.

The dataset that includes only captions yields much lower results than the dataset that includes both captions and comments. This is to be expected given the short length of the

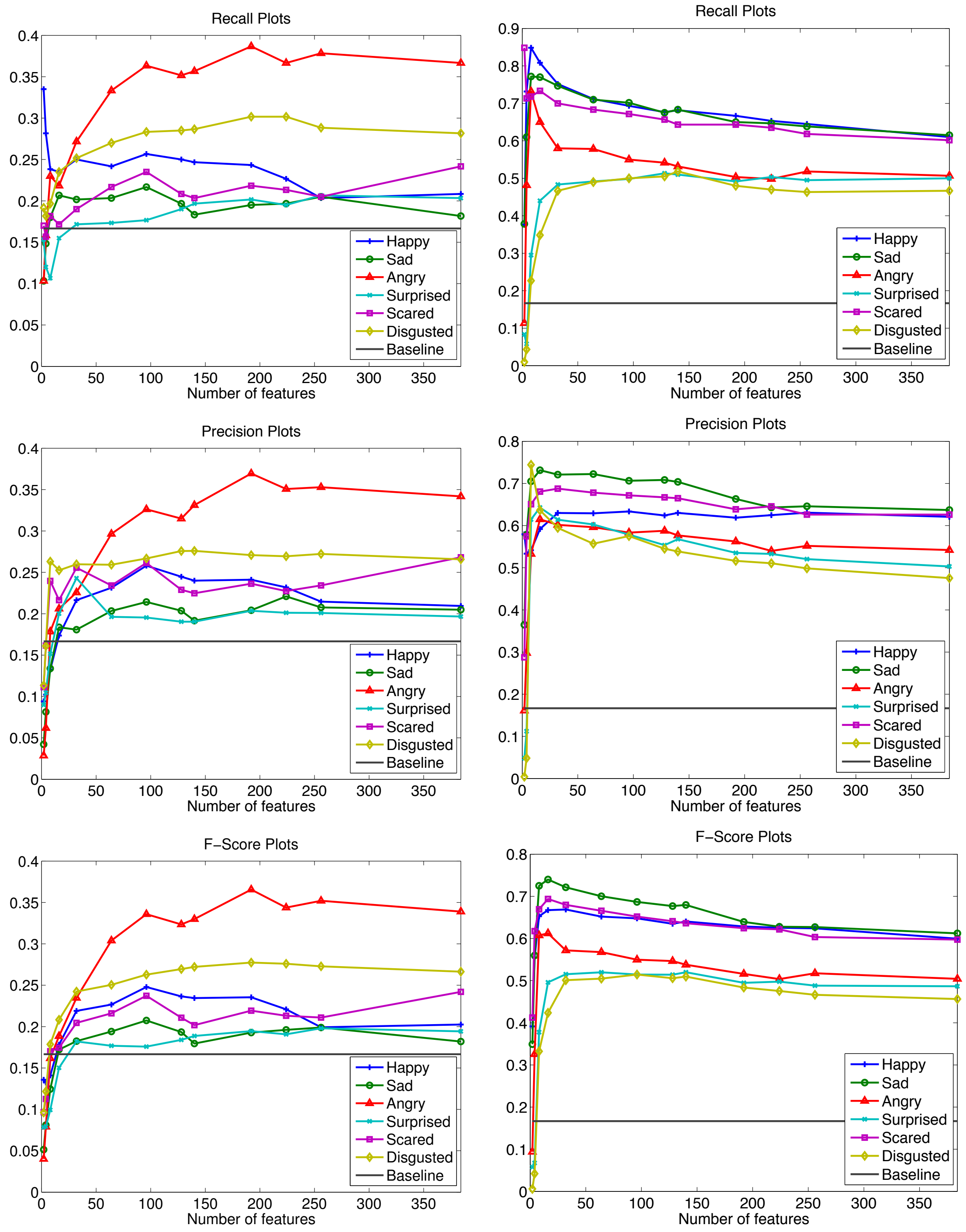


Figure 7.2: (Top) recall, (middle) precision and (bottom) F-score of the 6-class emotion classification problem over the (left) caption dataset and (right) caption+comment dataset

Position	**Captions**	**Captions+Comments**
1	red/NN	**negemo (LIWC)**
2	of/IN the/DT	<mood>/JJ
3	human/JJ	**anx (LIWC)**
4	brain/NN	**posemo (LIWC)**
5	dinosaur/NNS	**sad (LIWC)**
6	figure/NNS	**anger (LIWC)**
7	tank/NN	gross/JJ
8	while/IN the/DT	add/JJ
9	there/EX be/VBP several/JJ	mad/JJ
10	picture/NN there/EX be/VBZ	scary/JJ
11	while/IN	it/PP be/VBZ very/RB
12	hockey/NN mask/NN	creepy/JJ
13	**anger (LIWC)**	picture/NN because/IN it/PP
14	stand/VBP	**posScore (SWN)**
15	dinosaur/NNS be/VBP	lonely/JJ
16	hold/VBG	<mood>/JJ because/IN
17	white/JJ hockey/NN mask/NN	depressing/JJ
18	this/DT	strange/JJ
19	skull/NN	disturbing/JJ
20	**percept (LIWC)**	figure/NNS

Table 7.2: Features selected by mRMR. Bold fonts indicate LIWC and SentiWordNet (SWN) features; the rest of the features are TF-IDF features of n-grams (represented by lemma1/posTag1 . . . lemmaN/posTagN).

texts and the lack of emotion-related words in the dictionary; even with this limitation, however, the SVM slightly improved over the baseline, obtaining 26% accuracy using 96 features. Results for the full (caption and comment) dataset peaked at 62.5% accuracy using 16 features.

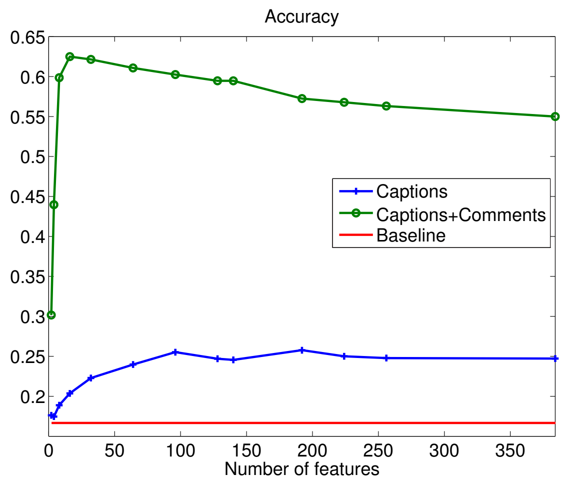


Figure 7.3: Accuracy of the 6-way emotion classification problem

Mood	Captions (96 features)			Caption+Comments (16 feat.)		
	Precision	**Recall**	**F-score**	**Precision**	**Recall**	**F-score**
Happy	0.26 ± 0.08	0.26 ± 0.10	0.25 ± 0.09	0.59 ± 0.04	0.81 ± 0.06	0.67 ± 0.04
Sad	0.21 ± 0.09	0.22 ± 0.07	0.21 ± 0.07	0.73 ± 0.04	0.77 ± 0.05	0.74 ± 0.04
Angry	0.33 ± 0.04	0.36 ± 0.05	0.34 ± 0.05	0.62 ± 0.05	0.65 ± 0.05	0.61 ± 0.03
Surprised	0.20 ± 0.06	0.18 ± 0.08	0.18 ± 0.06	0.64 ± 0.09	0.44 ± 0.09	0.50 ± 0.08
Scared	0.26 ± 0.08	0.24 ± 0.07	0.24 ± 0.07	0.68 ± 0.05	0.73 ± 0.06	0.69 ± 0.04
Disgusted	0.27 ± 0.04	0.28 ± 0.05	0.26 ± 0.04	0.63 ± 0.07	0.35 ± 0.09	0.42 ± 0.07
Accuracy		0.26 ± 0.02			0.63 ± 0.02	

Table 7.3: Mean and standard deviation of precision, recall, and F-score values for caption dataset and caption+comment dataset

Our method obtains good precision results over the full dataset, but the recall values for the *disgust* and *surprise* classes are low, which causes lower F-Score and accuracy values. This is apparent in Table 7.4, which shows an average confusion matrix from the sub-experiments that achieved 66% accuracy. Each row (from top to bottom) corresponds to instances labeled as *happiness*, *sadness*, *anger*, *surprise*, *fear* and *disgust*. Each column (from left to right) corresponds to instances predicted for the same classes in the same order.

	H	Sa	A	Su	F	D
Happiness	0.83	0.00	0.00	0.17	0.00	0.00
Sadness	0.33	0.67	0.00	0.00	0.00	0.00
Anger	0.17	0.00	0.83	0.00	0.00	0.00
Surprise	0.33	0.00	0.17	0.50	0.17	0.00
Fear	0.17	0.00	0.00	0.00	0.83	0.00
Disgust	0.00	0.17	0.00	0.33	0.17	0.33

Table 7.4: Average confusion matrix for a caption+comments run (66% accuracy)

The confusion matrix highlights how *disgust* and *surprise* examples are often misclassified. It also shows a tendency to assign *happiness* as the mood when making a mistake. Overall, the results show that our method of predicting emotion, using mRMR feature selection and 6-way SVM classification, is significantly superior to the baseline classifier.

7.3 Elicitation of Spatial Language

One of the main drawbacks of SpatialNet, which we introduced in Chapter 6, is that currently all of the components, including lexical units, spatial frames, and spatial vignettes, must be defined manually. One of our goals for future work on SpatialNet is to investigate ways to automatically or semi-automatically populate a SpatialNet for a new language. One technique that we hope to explore is an active learning algorithm that would use crowdsourcing to elicit descriptions of pictures representing spatial relations. These pictures could be automatically generated from spatial primitives using the WordsEye API we introduced in Chapter 5. In order to test the feasibility of using crowdsourcing as a part of the active learning pipeline, we conducted a pilot study in which we used Amazon Mechanical Turk to obtain simple descriptions of spatial configurations.

We used the WordsEye API to generate images from semantic representations of three spatio-graphic primitives: ON-TOP-SURFACE, RIGHT-OF, and IN-3D-ENCLOSURE. We then used Amazon Mechanical Turk to collect 5 English descriptions for each image by asking

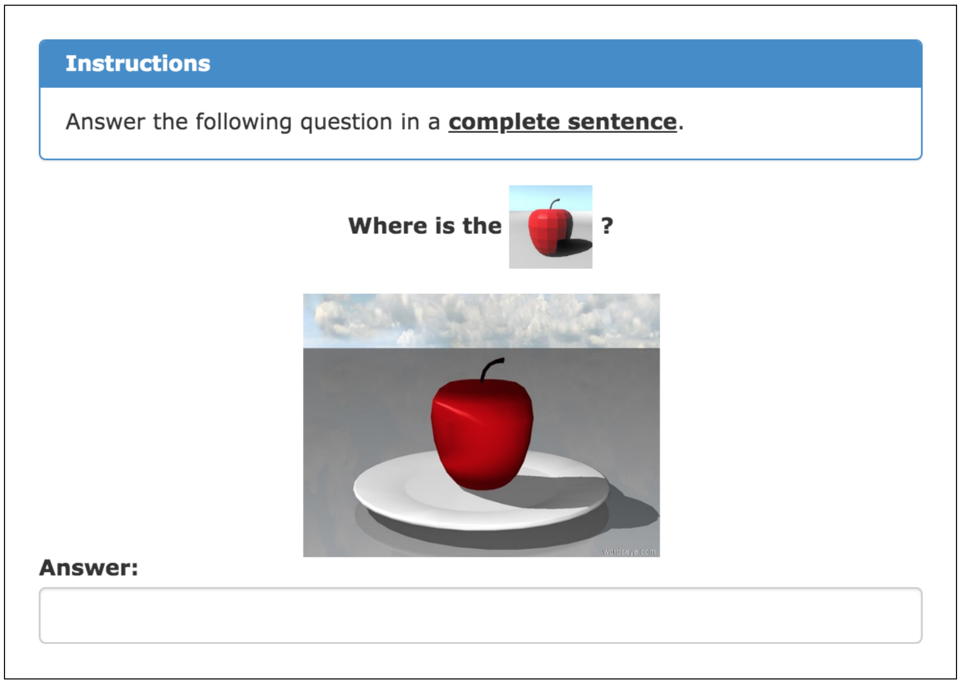


Figure 7.4: Example of spatial language elicitation HIT

questions of the form “Where is the X?”. Workers were instructed to answer the question in a complete sentence. An example of the HIT is shown in Figure 7.4. We restricted the task to workers who had a HIT approval rate greater than 98 and number of HITs approved greater than 500. To improve the likelihood that workers completing our task were fluent speakers of English, we restricted the task to workers who were located either in Great Britain or the United States. Workers were paid $0.04 for each completed HIT. The results are shown in Table 7.5.

We also designed and implemented a follow-up task in which we asked other AMT workers to judge whether the collected sentences were grammatical. For each of the last three descriptions in the third column of Table 7.5, we asked 3 workers to judge whether the given sentence was a correct sentence in English. We did not provide any additional instructions with the question. An example of this HIT is shown in Figure 7.5. We used the same qualification requirements as in the first task. Workers were paid $0.01 for each HIT completed. Results are shown in Table 7.6. We can see that for this small pilot study, a very simple sentence elicitation HIT in conjunction with this simple verification HIT can

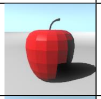
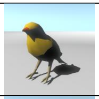
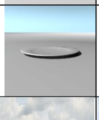
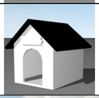
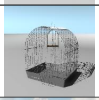
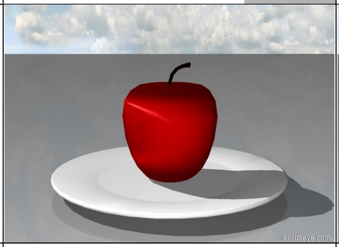
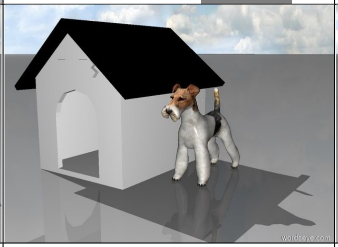
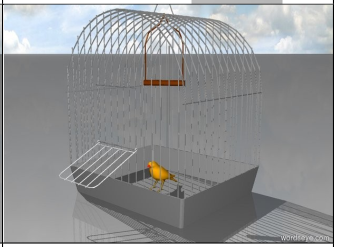
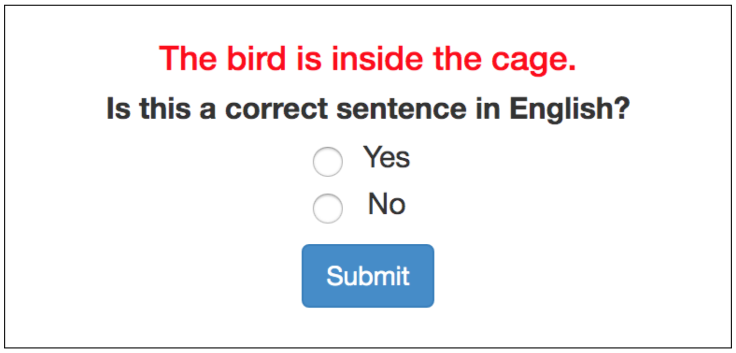
Relation	ON-TOP-SURFACE.R	RIGHT-OF.R	IN-3D-ENCLOSURE.R
Figure	APPLE.N 	DOG.N	BIRD.N 
Ground	DINNER-PLATE.N 	DOGHOUSE.N 	BIRDCAGE.N 
Scene			
Answers	1. apple is on a plate. 2. The apple is sitting in the middle of the white plate. 3. The red apple is sitting on a white plate. 4. The red apple is in the middle of a white plate. 5. The apple is on a plate.	1. The dog is standing next to the dog house. 2. The dog is standing beside the kennel. 3. The dog is standing outside of the doghouse. 4. The dog is next to the house. 5. The dog is next to the dog house.	1. The bird is sitting on the bottom of the metal birdcage. 2. The bird is on the bottom of the cage in the center. 3. The bird is inside the cage. 4. bird is in cage. 5. The yellow bird is standing on the bottom of its cage.

Table 7.5: Spatial descriptions collected with spatial language elicitation HIT

be successful at collecting grammatically correct descriptions of spatial relations. Although the pilot study was for English, the methodology could easily be adapted to other languages by obtaining translations for the instructions and questions, all of which are by intention kept as simple as possible.


The bird is inside the cage.

Is this a correct sentence in English?

Yes

No

Figure 7.5: Example of grammaticality verification HIT

Sentence	Yes	No
The bird is inside the cage.	3	0
bird is in cage.	0	3
The yellow bird is standing on the bottom of its cage.	3	0

Table 7.6: Counts of grammaticality judgments for each input sentence

7.3.1 Future Work: Active Learning of Spatial Language

One idea that we hope to explore in the future is to learn the SpatialNet for a language by utilizing an active learning algorithm in conjunction with crowdsourcing. The proposed methodology closely follows the workflow described in Chapter 3 for WELT, with the difference that the hypotheses will be developed automatically instead of manually by a field linguist. The core of documenting a language with WELT is based on eliciting descriptions of pictures in the target language, developing hypotheses as to which surface features (morphological, syntactic, and lexical) map to which semantic primitives, verifying hypotheses by showing native speakers machine-generated descriptions with the corresponding pictures, and refining hypotheses by generating more pictures and eliciting more descriptions. In WELT, the pictures used for elicitation are created manually by a trained field linguist. Likewise, hypotheses mapping surface features to semantic primitives are manually specified. In future work, we will investigate whether we can accomplish these tasks automatically using computational algorithms. These algorithms may also be used in combination with the existing WELT tools (e.g. a user would be able to manually refine automatically-proposed mappings).

The active learning technique we propose is based on eliciting descriptions of pictures representing spatial relations. The pictures will be automatically generated from semantic representations using the WordsEye API we introduced in Chapter 5. Textual descriptions will be elicited through crowdsourcing or may be elicited locally from native speakers. The system will use the pairings of descriptions and semantic representations to automatically develop hypotheses as to which surface linguistic features (morphological, syntactic, and/or lexical) map to which semantic primitives, to verify hypotheses by showing native speakers proposed descriptions of new pictures, and to refine hypotheses by generating more pictures and eliciting more descriptions.

We plan to use the Amazon Mechanical Turk API so that the system can automatically publish HITs and retrieve results during the learning process. We have demonstrated with our pilot study that it is possible to use Amazon Mechanical Turk to collect simple descriptions of 3D objects participating spatial relations, of the kind that would be needed for our active learning system. Without a trained field linguist guiding the elicitation session, we will need to develop other strategies to encourage informants to focus on particular aspects of a scene. In addition to displaying a single picture at a time, as in our pilot study, we anticipate that we will need to experiment with providing informants with two or more images in order to ensure that the desired semantic contrasts are captured in the descriptions. The system would generate pictures from semantic structures differing in some aspect and ask the annotator to provide a caption for each that would allow someone to distinguish between the two pictures.

For the verification stage of the learning process, we plan to have our system start with the underlying semantics and generate both a picture using WordsEye and a textual description using the currently hypothesized theory. Here, as when eliciting descriptions, we plan to experiment with different ways to obtain the judgments. The simplest method will be to show a single description with a single picture and ask the informant whether this is a valid description of the picture (yes/no). As an alternative, we plan to show multiple descriptions with a single picture and ask the informant to select which, if any, would be a valid description of the picture. Likewise, we could show a single description with multiple pictures and ask the informant to select which picture(s), if any, are indicated by the description.

One of the major benefits of using an automatic system for learning rather than working directly with a native speaker or obtaining linguist annotations is that the system can easily obtain a large amount of data, through generation of images and crowdsourcing. However, the size of our search space makes a brute force approach undesirable. While the number of SGPs and the size of the ontology is finite, every SGP has multiple parameters and/or constraints and every semantic category in the ontology has many affordances and functional properties. We do not know which of these will end up being important in a particular language. Therefore, efficient algorithmic techniques are required, and a large part of continuing research on this topic will need to be determining the algorithms that are most effective for this task. Here, we sketch a few possibilities that may be considered when experimenting with algorithms.

The goal in any algorithm is to track down minimal pairs: that is, two instances that differ in only one aspect of their semantics and/or surface language. This is easy to do when lexical information maps one-to-one with SGPs. For instance, *the ball is* ***on*** *the table* and *the ball is* ***under*** *the table* map to the SGPs ON-UPWARD-SURFACE and UNDER-CANOPY, respectively. It is trivial in this case to hypothesize that *on* maps to ON-UPWARD-SURFACE and *under* maps to UNDER-CANOPY. Examples such as those given in Figure 6.2 are more interesting (and more difficult to learn) because the mapping involves properties of the objects as well. The algorithm will need to vary SGPs (and parameters of SGPs) as well as objects participating in the spatial relation. One approach could be to select a fixed pair of objects and vary the SGP. As an alternative method, one could choose a single SGP and vary the objects. We anticipate that the most effective algorithm will use a combination of these approaches.

7.4 Elicitation of Imaginative Language

In this section,[2] we describe how we use the semantic grounding provided by VigNet to elicit imaginative language using crowdsourcing. Typically, in order to obtain visually-oriented language using crowdsourcing, researchers rely on pre-existing sets of images and collect descriptions of these. The crowdsourcing tasks described in Section 7.3 and Section 7.2 are two examples of these. However, there are situations in which obtaining sentences that describe new and possibly previously unimagined scenarios would be preferred. One example of such a situation is to evaluate a system's ability to produce an accurate illustration of a sentence. Illustrations of sentences can be useful in many applications, including creating story boards for movie scripts and creating picture books for children. For people without the skill to paint or draw illustrations themselves, it is necessary to find relevant pictures in other ways. It is possible that users will seek to illustrate text that describes previously unseen situations, especially if they are interested in fantasy and science fiction genres, so a corpus intended to evaluate an automatic illustration system would ideally include some imaginative sentences.

The corpus described in this section was used to evaluate WordsEye's ability to produce illustrations for imaginative sentences, as compared to Google image search. Standard image search engines are limited to pictures that already exist in their databases, biasing them toward retrieving images of mundane and real-world scenarios. In contrast, a scene generation system like WordsEye can illustrate a much wider range of images, allowing users to visualize unusual and fantastical scenes. We saw some examples of imaginative scenes created in WordsEye in the examples from the emotional language elicitation HIT, shown in Figure 7.1. Other examples of imaginative scenes that have been created in WordsEye are shown in Figure 7.6.

7.4.1 Crowdsourcing Methodology

We used Amazon Mechanical Turk to obtain imaginative sentences. We gave AMT workers short lists of words divided into several categories and asked them to write a short sentence

[2]Some of the material in this section was previously published in Ulinski *et al.* [2018].

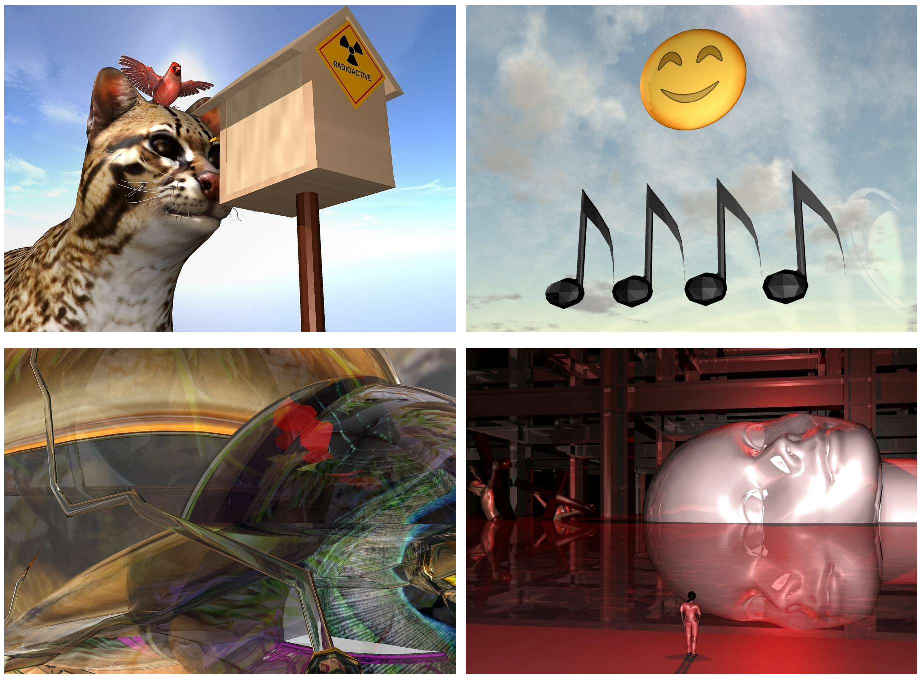


Figure 7.6: Imaginative images: situational, iconic, abstract, fantastic

using at least one word from each category. The words provided to the workers represent objects, properties, and relations supported by WordsEye, and were extracted from among the lexicalized concepts contained in VigNet.

To help workers construct sentences of different types, we organized the objects, properties, and relations into a few basic categories. The categories are listed in Table 7.7. We restricted the lexicon to include only commonly known words that could be easily understood and recognized visually. We excluded super-types such as “invertebrate” and sub-types such as “european elk”. We omitted obscure terms such as “octahedron” or “diadem”. The resulting lexicon included about 1500 terms and phrases.

We created 12 different combinations of categories with 20 HITs per combination. Each HIT randomly presented different words for each category in order to elicit different types of sentences from the workers. This involved varying the types and number of categories as well as the order of the items in the categories. We wanted to encourage sentences such as *There is a blue dog on the large table* as well as different orders and constructs like *The dog on the large table is blue*. Each HIT showed 4 or 5 categories, with three words per

Category	**Definition**	**Examples**
PROP	small objects that could be held or carried	*cellphone, apple, diamond*
FIXTURE	large objects such as furniture, vehicles, plants	*couch, sailing ship, oak tree*
ANIMAL	animals	*dolphin, chicken, llama*
SPATIAL TERM	terms representing spatial relations	*above, against, facing, on*
NUMBER	small numbers	*one, four, nine, twelve*
COLOR	common colors	*beige, green, scarlet, black*
SIZE	general size or specific dimensions	*big, tiny, thin, 5 feet long*
DISTANCE	distances	*4 inches, five meters, 10 feet*
SURFACE PROPERTY	properties of surfaces	*opaque, shiny, transparent*
LOCATION	terms representing terrain types and locations	*field, driveway, lake, forest*
BUILDING	buildings and architectural structures	*doghouse, castle, skyscraper*

Table 7.7: Categories of words in the lexicon, used to elicit imaginative sentences

category. Table 7.8 shows all the combinations of categories.

Our instructions specified that workers write a single sentence using a maximum of 12 words. Words could be in any order as long as the resulting sentence was grammatical. We allowed the use of any form of a given word; for example, using a plural noun instead of a singular. We also allowed the use of *filler words* not listed in the categories, but asked workers not to add any unlisted *content words*. We defined *filler words* as words with "little meaning on their own, but that are used to make the sentence grammatical (e.g. *the*, *has*, *is*, *with*)" and *content words* as words that "refer to an object, action, or characteristic (e.g. *eat*, *shallow*, *organization*)." An example HIT is shown in Figure 7.7.

We restricted our task to workers who had completed at least 100 HITs previously with an approval rate of at least 98%. We paid $.04 per assignment. We started with 240 unique

AMT Column Headings	**WordsEye Lexical Categories**	
1. Noun1, Noun2, Spatial Term, Adjective	Noun1 is PROP. Noun2 is FIXTURE.	Spatial Term is SPATIAL TERM. Adjective is SIZE, COLOR or SURFACE PROPERTY. Distance is DISTANCE. Location is BUILDING or LOCATION. Size is SIZE. Color is COLOR. Number is NUMBER.
2. Adjective, Noun1, Noun2, Spatial Term		
3. Adjective, Noun1, Noun2, Spatial Term	Noun1 is ANIMAL. Noun2 is FIXTURE.	
4. Noun1, Noun2, Spatial Term, Distance, Adjective	Noun1, Noun2 are PROP, FIXTURE or ANIMAL.	
5. Noun1, Noun2, Spatial Term, Location	Noun1 is ANIMAL. Noun2 is PROP or FIXTURE.	
6. Noun1, Noun2, Spatial Term, Distance, Adjective	Noun1, Noun2 are PROP or FIXTURE .	
7. Adjective, Noun1, Noun2, Spatial Term, Distance		
8. Noun1, Noun2, Spatial Term, Location		
9. Noun1, Noun2, Spatial Term, Color, Size		
10. Noun1, Noun2, Spatial term, Number		
11. Noun1, Noun2, Spatial term, Number, Adjective		
12. Adjective, Noun1, Noun2, Spatial Term, Number		

Table 7.8: Possible combinations of categories for the imaginative sentence construction task. Examples of sentences collected for each combination are included in Table 7.9.

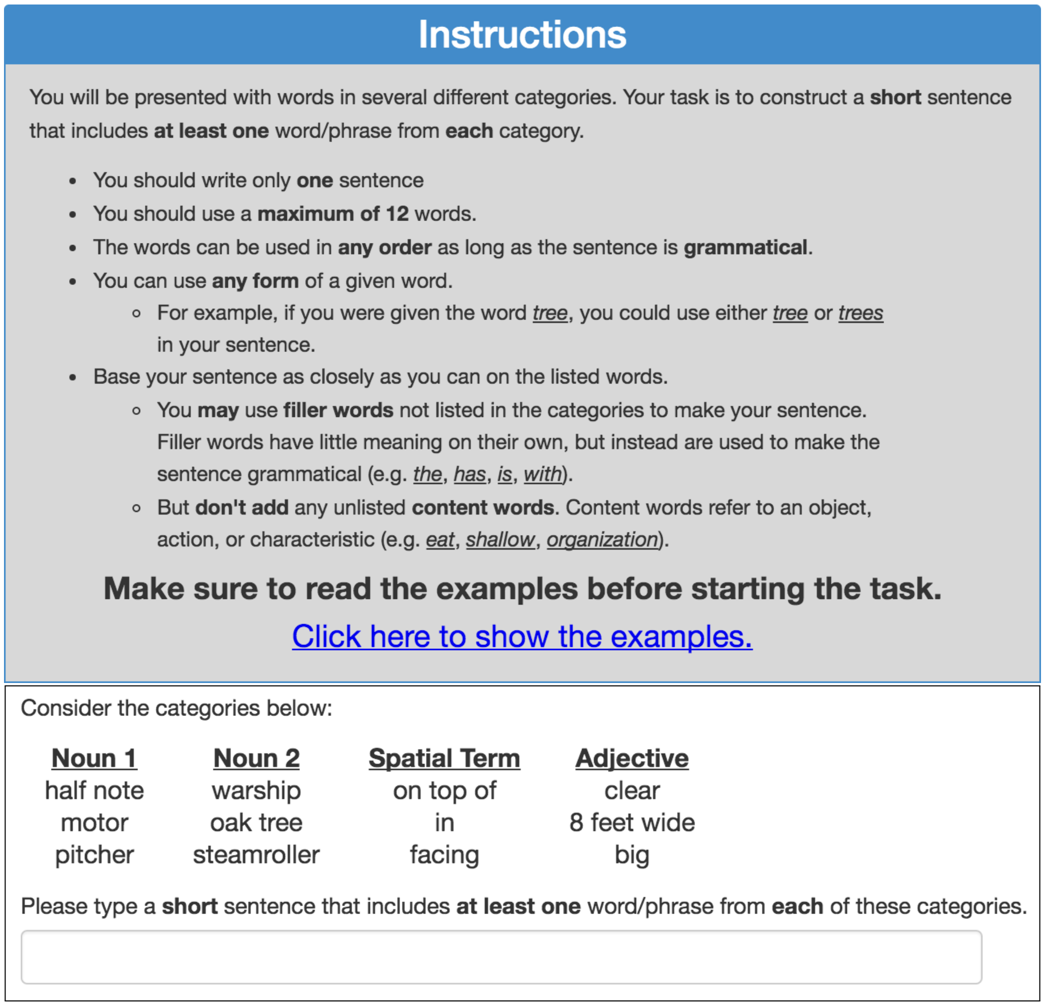
.

combinations of words and collected one sentence for each of these. After filtering out ungrammatical sentences, we ended up with a total of 209 imaginative sentences. Table 7.9 shows examples of sentences we collected for each of the category combinations in Table 7.8.

7.4.2 Evaluating WordsEye: Imaginative and Realistic Sentences

We used the imaginative sentences we collected as part of an evaluation of the WordsEye system.[3] Specifically, we evaluated WordsEye's ability to create a picture that illustrates a sentence, as compared to traditional image search methods. In addition to the set of

[3]For a more detailed description of this evaluation, see Ulinski *et al.* [2018].



Instructions

You will be presented with words in several different categories. Your task is to construct a **short** sentence that includes **at least one** word/phrase from **each** category.

- You should write only **one** sentence
- You should use a **maximum of 12** words.
- The words can be used in **any order** as long as the sentence is **grammatical**.
- You can use **any form** of a given word.
  - For example, if you were given the word *tree*, you could use either *tree* or *trees* in your sentence.
- Base your sentence as closely as you can on the listed words.
  - You **may** use **filler words** not listed in the categories to make your sentence. Filler words have little meaning on their own, but instead are used to make the sentence grammatical (e.g. *the*, *has*, *is*, *with*).
  - But **don't add** any unlisted **content words**. Content words refer to an object, action, or characteristic (e.g. *eat*, *shallow*, *organization*).

Make sure to read the examples before starting the task.

Click here to show the examples.

Consider the categories below:

Noun 1	Noun 2	Spatial Term	Adjective
half note	warship	on top of	clear
motor	oak tree	in	8 feet wide
pitcher	steamroller	facing	big

Please type a **short** sentence that includes **at least one** word/phrase from **each** of these categories.

Figure 7.7: Example of imaginative sentence collection HIT

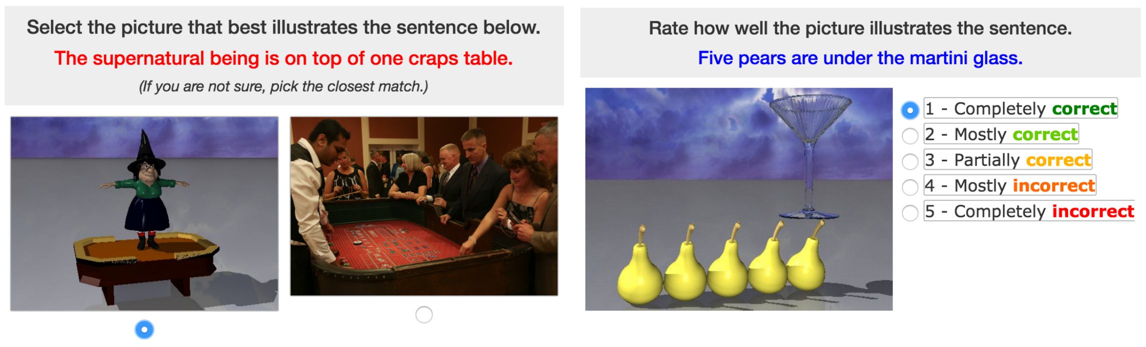
imaginative sentences, we used a set of realistic sentences extracted from the PASCAL image caption corpus [Rashtchian *et al.*, 2010]. For each imaginative and realistic sentence, we compared the highest ranking pictures found using Google search to those produced by WordsEye. To obtain potential illustrations, we used each sentence as a search query on Google image search. We presented the top four search results to workers on Amazon Mechanical Turk and asked them to select which picture best illustrated the sentence. We selected the best Google image among the four based on the crowdsourced majority vote. Likewise, we used WordsEye to generate four images for each sentence, varying the camera angle and choice of 3D objects in each case. Again, we used a crowdsourced majority vote

Example imaginative Sentences

1. $\textit{The } \underset{\text{Noun1}}{\underline{\textit{book}}} \ \underset{\text{Spatial}}{\underline{\textit{near}}} \textit{ the } \underset{\text{Noun2}}{\underline{\textit{oak tree}}} \textit{ is } \underset{\text{Adj.}}{\underline{\textit{white}}}\textit{.}$
2. $\underset{\text{Spatial}}{\underline{\textit{On}}} \textit{ the } \underset{\text{Noun1}}{\underline{\textit{couch}}} \textit{ is a } \underset{\text{Adj.}}{\underline{\textit{gray}}} \ \underset{\text{Noun2}}{\underline{\textit{graduation cap}}}\textit{.}$
3. $\textit{The } \underset{\text{Adj.}}{\underline{\textit{large}}} \ \underset{\text{Noun1}}{\underline{\textit{prawn}}} \textit{ is } \underset{\text{Spatial}}{\underline{\textit{on top of}}} \textit{ the } \underset{\text{Noun2}}{\underline{\textit{stool}}}\textit{.}$
4. $\textit{A } \underset{\text{Adjective}}{\underline{\textit{maroon}}} \ \underset{\text{Noun1}}{\underline{\textit{coffee cup}}} \textit{ is } \underset{\text{Distance}}{\underline{\textit{three feet}}} \ \underset{\text{Spatial}}{\underline{\textit{behind}}} \textit{ the } \underset{\text{Noun2}}{\underline{\textit{alligator}}}\textit{.}$
5. $\underset{\text{Spatial term}}{\underline{\textit{On top of}}} \textit{ the } \underset{\text{Location}}{\underline{\textit{office}}} \ \underset{\text{Noun2}}{\underline{\textit{bunk bed}}} \textit{ was a } \underset{\text{Noun1}}{\underline{\textit{possum}}}\textit{.}$
6. $\textit{The } \underset{\text{Adj.}}{\underline{\textit{brown}}} \ \underset{\text{Noun1}}{\underline{\textit{shovel}}} \textit{ is } \underset{\text{Distance}}{\underline{\textit{three feet}}} \ \underset{\text{Spatial}}{\underline{\textit{behind}}} \textit{ the } \underset{\text{Noun2}}{\underline{\textit{flower}}}\textit{.}$
7. $\textit{The } \underset{\text{Adj.}}{\underline{\textit{tiny}}} \ \underset{\text{Noun1}}{\underline{\textit{toaster}}} \textit{ is } \underset{\text{Distance}}{\underline{\textit{10 feet}}} \ \underset{\text{Spatial term}}{\underline{\textit{in front of}}} \textit{ the } \underset{\text{Noun2}}{\underline{\textit{bird cage}}}\textit{.}$
8. $\underset{\text{Spatial term}}{\underline{\textit{In front of}}} \textit{ the } \underset{\text{Location}}{\underline{\textit{palace's}}} \ \underset{\text{Noun1}}{\underline{\textit{oven}}} \textit{ is a } \underset{\text{Noun2}}{\underline{\textit{napkin}}}\textit{.}$
9. $\textit{The } \underset{\text{Size}}{\underline{\textit{tiny}}} \ \underset{\text{Color}}{\underline{\textit{blue}}} \ \underset{\text{Noun1}}{\underline{\textit{sailing ship}}} \textit{ is } \underset{\text{Spatial}}{\underline{\textit{above}}} \textit{ the } \underset{\text{Noun2}}{\underline{\textit{crib}}}\textit{.}$
10. $\textit{The } \underset{\text{Noun1}}{\underline{\textit{spatula}}} \textit{ was } \underset{\text{Spatial}}{\underline{\textit{close to}}} \ \underset{\text{Num.}}{\underline{\textit{eight}}} \ \underset{\text{Noun2}}{\underline{\textit{eggs}}}\textit{.}$
11. $\underset{\text{Num.}}{\underline{\textit{Five}}} \ \underset{\text{Adjective}}{\underline{\textit{magenta}}} \ \underset{\text{Noun1}}{\underline{\textit{cantaloupes}}} \textit{ are } \underset{\text{Spatial}}{\underline{\textit{in}}} \textit{ the } \underset{\text{Noun2}}{\underline{\textit{trash can}}}\textit{.}$
12. $\textit{There were } \underset{\text{Num.}}{\underline{\textit{seven}}} \textit{ pieces of } \underset{\text{Noun1}}{\underline{\textit{celery}}} \ \underset{\text{Spatial}}{\underline{\textit{on}}} \textit{ the } \underset{\text{Adj.}}{\underline{\textit{gray}}} \ \underset{\text{Noun2}}{\underline{\textit{computer printer.}}}$

Table 7.9: Examples of imaginative sentences collected with AMT. Table 7.8 contains definitions for the column headings.

to select the best WordsEye image. We then evaluated the best Google image versus the best WordsEye image in two ways. First, we used a simple comparison task, showing AMT workers both images and asking which one best illustrated the sentence. Second, we created a numerical rating task, asking AMT workers to rate how well each picture illustrated the sentence, from 1 (completely correct) to 5 (completely incorrect). Examples of the Amazon



Select the picture that best illustrates the sentence below.

The supernatural being is on top of one craps table.

(If you are not sure, pick the closest match.)

(a)

Rate how well the picture illustrates the sentence.

Five pears are under the martini glass.

[✓ 1 - Completely correct]
2 - Mostly correct
3 - Partially correct
4 - Mostly incorrect
5 - Completely incorrect

(b)

Figure 7.8: Examples of the WordsEye evaluation AMT tasks: (a) image comparison task[4] and (b) rating task.

Mechanical Turk HITs for these two evaluation tasks are shown in Figure 7.8.

For the first crowdsourcing task, we found that for imaginative sentences, pictures produced by WordsEye were preferred, but for realistic sentences, Google Image Search results were preferred. For the second crowdsourcing task, WordsEye pictures had an average rating of 2.58 on imaginative sentences and 2.54 on realistic sentences (about halfway between "mostly correct" and "partially correct" in both cases). While Google search did perform better than WordsEye on realistic sentences, its performance breaks down when faced with imaginative sentences. Google images had an average rating of 1.87 on realistic sentences (between "completely correct" and "mostly correct") and an average rating of 3.82 on imaginative sentences (between "partially correct" and "mostly incorrect"). The complete listing of number of votes and numerical ratings received by Google and WordsEye images for each sentence can be found in Coyne [2017, Appendix A].

Our evaluation showed WordsEye to be superior for imaginative sentences and Google image search to be superior for realistic sentences. While this overall result is not unexpected, our work allows us to quantify precisely what the gap in performance is. In particular, while the average rating of WordsEye on realistic sentences was just 0.665 below that of Google, WordsEye's ratings on imaginative sentences was 1.244 higher than

[4]Google image source: `https://en.wikipedia.org/wiki/Craps`.

Google's. This suggests that as WordsEye and text-to-scene technology in general improve, they may become a viable alternative to image search even for realistic sentences, but that it might be difficult to adapt traditional image search techniques to retrieve illustrations for imaginative sentences. Creativity is something that too often gets overlooked in technology development, and our results show that research into text-to-scene generation could play an important role in addressing the issue. Our new corpus of imaginative sentences may also have applications for other researchers studying language in a visual context or those interested in spatial language in general.

7.5 Conclusion

In this chapter, we have described three ways that text-to-scene generation can be used to elicit language data via crowdsourcing. First, we showed how we used 3D scenes created in WordsEye to elicit a corpus of descriptions containing emotional language. Next, we demonstrated how we can use the WordsEye API for generating scenes from semantic primitives to collect simple spatial descriptions. These spatial descriptions can potentially be used as part of an active learning algorithm in the future. Finally, we showed how we can use lexical information from VigNet to elicit descriptions of imaginative scenarios. Overall, we have demonstrated that text-to-scene generation is an extremely flexible tool that can be applied to elicit many kinds of language data. Although the crowdsourcing examples described in this chapter all elicit English text, the techniques we describe could easily be applied to other languages, provided there are a sufficient number of native speakers of that language. For the elicitation of emotional and spatial language, we would simply need to translate the instructions given to the workers. The lexicon used to elicit imaginative sentences could be generated for another language using the kind of VigNet–WordNet mapping we described in Chapter 6 and used to obtain a German lexicon.

Chapter 8

Conclusion

8.1 Summary of Contributions

In this thesis, we have shown how text-to-scene generation can be applied to facilitate language elicitation and documentation.

The first major contribution of this thesis is that we have used text-to-scene generation as a tool for linguistic fieldwork. The WordsEye Linguistics Tools (WELT) assist field linguists with the elicitation and documentation of endangered languages. With WELT, linguists use text-to-scene-generation to create custom elicitation materials in the form of 3D scenes. These scenes are used to elicit descriptions from native speaker informants. Along with providing the means to formally document the semantics of a language, a function largely absent from other existing field linguistics tools, the WELT documentation tools also result in the creation of a text-to-scene system for the endangered language. This text-to-scene system can be used to verify theories with the informant. Although some parts of WELT are still in development, the prototypes we have created and used to elicit and document Nahuatl and Arrernte show the usefulness of text-to-scene generation for this application. Having a reasonably accurate syntactic parser for the endangered language is an important component of the text-to-scene system that is created with WELT. To this end, we have performed experiments that show that incrementally learning a dependency parser based on a small number of examples can assist with the annotation of dependency structures. This demonstrates that this method could be a feasible way to acquire a syntactic parser for

WELT, in a more linguist-friendly way than using LFG or another grammar formalism. It would also be useful as part of a tool for annotating the syntax of endangered language data independently of WELT. We have also adapted WordsEye into a system that can produce a 3D scene directly from a semantic representation, and used this system to modify the user interface for creating 3D scenes in WELT to allow users to directly manipulate the underlying semantics of a scene.

Another significant contribution of this thesis is to provide a framework for the support of multilingual text-to-scene generation. The text-to-scene system for an endangered language created with WELT is one example of this. In addition, we have paved the way for text-to-scene systems in other languages by providing a language-independent semantic interface to the graphics generation component of WordsEye. The ability to generate a 3D scene directly from primitive semantic relations means that a text-to-scene system for a language can be created by mapping text to its underlying semantics and then using the 3D graphics functionality provided by WordsEye to generate a 3D scene. We have also introduced a new multilingual resource, SpatialNet, that uses frame semantics to link linguistic expressions to spatial and graphical primitives, and demonstrated a methodology that uses SpatialNet in conjunction with other existing NLP tools to produce a text-to-scene system for a language. We anticipate that SpatialNet will be useful outside of text-to-scene generation as well, for example to improve machine translation of spatial relations or to identify spatial relations and roles in text more precisely than is possible with the spatial annotation frameworks currently in general use.

8.1.1 Tools and Resources

The work done for this thesis has resulted in a number of resources that will be of use in future research, both in continuing the work described in this thesis and for researchers in related fields. We list some of these resources here:

The WordsEye Linguistics Tools (WELT): The tools we created for elicitation and documentation may be used in the future to study other endangered languages. WELT English provides a field linguist with tools for building and conducting elicitation sessions based on sets of custom 3D scenes. WELT L2 provides a way to formally document the lexical semantics of an endangered language. Formal hypotheses can be verified using a text-to-scene system that takes input in the endangered language, analyzes it based on the formal model, and generates a picture representing the meaning. Our prototype of WELT L2 will require further development before it can be released, but the WELT English elicitation tools are currently useable (as demonstrated by our elicitation of Nahuatl topological relations) and will be further improved by integration with the latest version of WordsEye.

Arrernte-specific 2D and 3D content: We compiled a set of 2D and 3D models relevant to Arrernte culture and to the study of topological relations. In particular, we enhanced a set of black-and-white illustrations from the Eastern and Central Arrernte Picture Dictionary [Broad, 2008], adding color and cropping the background to make them suitable as 2D cutouts in WordsEye. In addition to their usefulness in extending WordsEye's relevance to other cultures and geographic regions, the models and images may be of interest to others studying Australian indigenous languages or for the purpose of outreach and education.

Topological relations 3D scenes and Nahuatl descriptions: We created a set of 3D scenes representing basic topological relations and used these to elicit descriptions from a native speaker of Nahuatl, which we also translated and glossed. This data, along with the audio recordings of the elicitation sessions with our Nahuatl informant, may be used in future studies in Nahuatl linguistics.

Annotated Arrernte data: We compiled a set of Arrernte sentences, primarily from Wilkins [1989] and Broad [2008], and translated and glossed these using SIL FieldWorks. In addition, we added to the FieldWorks project the phonetic and phonological information needed for the program to produce a morphological parser. The sentences we chose were interesting either because of spatial language or the use of case marking. This data will be useful in the future for linguists interested in spatial language and case in Arrernte.

Multilingual spatial relation and motion treebank: We created a new corpus of English, Spanish, German, and Egyptian Arabic descriptions of spatial relations and motion events, which we annotated with syntactic dependency structures and other linguistic information. This corpus and treebank will be useful for researchers interested in the syntax of spatial language across languages.

Deep semantic representation of spatial relations and SpatialNet: Although the SpatialNet resources we are developing for English and German are still in development, the SpatialNet framework and deep semantic representation is a useful tool that can be applied to other languages. The combination of multilingual SpatialNet resources with our pipeline for text-to-scene generation with SpatialNet is a significant step toward multilingual text-to-scene generation. In addition, we anticipate SpatialNet will be useful for other natural language processing applications that can make use of spatial language understanding, such as machine translation and spatial role labeling. Comparing the SpatialNet resources created for different languages may also be of interest to those working in linguistic typology.

Crowdsourced corpora of emotional and imaginative language: We used crowdsourcing to annotate a collection of images from the WordsEye gallery with high-level descriptions, a mood label, and an explanation of why the mood label was chosen. In addition to its potential use for training a system to automatically assign a mood to WordsEye scenes, this corpus will be useful for other researchers interested in automatically detecting emotion from text. We also used crowdsourcing to collect a set of imaginative sentences covering a wide range of graphical scenarios. This corpus may be of interest to other researchers working at the intersection of vision and language.

8.2 Future Work

We have discussed possibilities for future work throughout this thesis. In this section, we briefly summarize some of the main areas for future work.

Integration of a “linguist-friendly” syntactic parser with WELT: Currently WELT relies on the existence of grammars created with external tools, such as XLE, to handle syntactic processing. This requires WELT users to have knowledge of such tools and of grammar formalisms. Ideally, WELT would include a user interface for annotating sentences with dependency structures to allow the "stealthy" construction of a syntactic parser in the style of SIL Fieldworks Language Explorer. The parser would be learned using the methods we discussed in Chapter 4. Further exploration of other parsing methods, including methods of adapting existing parsers to new languages, and ways to incorporate syntactic properties manually specified by the user into the parser, would also be a useful direction for research.

Integration of WELT with the WordsEye web API and SpatialNet: As we discussed in Chapter 5, there are a number of advantages to using the latest version of WordsEye instead of the Mac OS X desktop application, and these advantages will only grow as more functionality is added to the WordsEye web application. We have begun the process of updating WELT by creating the pipeline to generate 3D scenes from semantic primitives and using this to develop a new user interface for creating elicitation materials in WELT. Continuation of this work will incorporate the web API into other parts of the WELT tools, including developing user interfaces for adding custom content that use the WordsEye API to add custom 2D cutouts and backdrops to a user's WordsEye account. The WordsEye developers are also working on streamlining the process for adding custom 3D models and for linking user content to semantic types in the ontology, both of which will be useful for WELT. We will also update the WELT documentation tools to use SpatialNet for the underlying semantic representation rather than the English VigNet, providing a more language-independent representation of lexical semantics. Since the SpatialNet text-to-scene generation pipeline already uses the WordsEye web API to generate 3D scenes, this will also allow WELT to use the latest version of WordsEye for verifying grammars by generating scenes from L2 text.

Extension of WELT to include automatic learning algorithms: Currently, WELT is designed for a field linguist to create their own 3D scenes, work in person with a native speaker informant to elicit descriptions of the scenes, hypothesize theories for the semantics, formally document the observed semantics, and use the resulting text-to-scene system to verify theories with the informant. This in turn leads the linguist to create further elicitation materials, and the formal documentation is refined as the process continues. In future work, we hope to supplement this manual workflow with automatic algorithms, using active learning techniques. Using SpatialNet as the representation for the formal documentation in WELT L2 will allow us to incorporate some of the methods proposed in Chapter 7, Section 7.3.1.

Bibliography

Aboriginal Australian Art & Culture - Alice Springs. `http://aboriginalart.com.au/`.

Giovanni Adorni, Mauro Di Manzo, and Fausto Giunchiglia. Natural Language driven Image Generation. In *Proceedings of the 10th International Conference on Computational Linguistics and 22nd Annual Meeting of the Association for Computational Linguistics*, pages 495–500, Stanford, California, USA, July 1984. Association for Computational Linguistics.

Amazon Mechanical Turk, Inc. Amazon Mechanical Turk. `https://www.mturk.com/`, 2018.

Felix Ameka, Carlien de Witte, and David P. Wilkins. Picture series for positional verbs: Eliciting the verbal component in locative descriptions. In David P. Wilkins, editor, *Manual for the 1999 Field Season*, pages 48–54. Max Planck Institute for Psycholinguistics, Nijmegen, 1999.

Jonathan D. Amith. Nahuatl Learning Environment. `http://www.balsas-nahuatl.org/`.

Neil Ashton. Nahuatl morphology revisited. `http://www.speechlike.org/2013/03/nahuatl-morphology-revisited/`, March 2013.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, 2010. European Languages Resources Association (ELRA).

Norman Badler, Ramamani Bindiganavale, Juliet Bourne, Martha Palmer, Jianping Shi, and William Schuler. A Parameterized Action Representation for Virtual Human Agents.

In *Embodied Conversational Agents*, pages 256–284. MIT Press, Cambridge, MA, January 2000.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet Project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada, August 1998. Association for Computational Linguistics.

Collin Baker. FrameNet, Present and Future. In *The First International Conference on Global Interoperability for Language Resources*, pages 12–17, 2008.

Stephen Beale and Tod Allman. Linguist's Assistant: A Resource For Linguists. In *Proceedings of the 9th Workshop on Asian Language Resources*, pages 41–49, Chiang Mai, Thailand, November 2011. Asian Federation of Natural Language Processing.

Kenneth R. Beesley and Lauri Karttunen. Finite State Morphology. `http://www.fsmbook.com`, 2003.

Emily M. Bender, Dan Flickinger, and Stephan Oepen. The Grammar Matrix: An Open-Source Starter-Kit for the Rapid Development of Cross-linguistically Consistent Broad-Coverage Precision Grammars. In *COLING-02: Grammar Engineering and Evaluation*. Association for Computational Linguistics, 2002.

Steven Bird and David Chiang. Machine Translation for Language Preservation. In *Proceedings of COLING 2012: Posters*, pages 125–134, Mumbai, India, December 2012. The COLING 2012 Organizing Committee.

Steven Bird. Natural Language Processing and Linguistic Fieldwork. *Computational Linguistics*, 35(3):469–474, September 2009.

Cheryl A Black and H Andrew Black. PAWS: Parser and writer for syntax: Drafting syntactic grammars in the third wave. In *SIL Forum for Language Fieldwork*, volume 2, 2009.

Cheryl A. Black and H. Andrew Black. Grammars for the people, by the people, made easier using PAWS and XlingPaper. In *Electronic Grammaticography*, pages 103–128. University of Hawai'i Press, Honolulu, October 2012.

H. Andrew Black and Gary F. Simons. The SIL FieldWorks Language Explorer Approach to Morphological Parsing. In *Texas Linguistics Society 10: Computational Linguistics for Less-Studied Languages*, 2008.

Bernd Bohnet. Very High Accuracy and Fast Dependency Parsing is not a Contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 89–97, Beijing, China, August 2010. COLING 2010 Organizing Committee.

Melissa Bowerman and Soonja Choi. Space under Construction: Language-Specific Spatial Categorization in First Language Acquisition. In *Language in Mind: Advances in the Study of Language and Thought*, pages 387–428. MIT Press, Cambridge, MA, US, 2003.

Melissa Bowerman and Eric Pederson. Topological relations picture series. In Stephen C. Levinson, editor, *Space Stimuli Kit 1.2*, page 51. Max Planck Institute for Psycholinguistics, Nijmegen, 1992.

Neil Broad. *Eastern and Central Arrernte Picture Dictionary.* IAD Press, Alice Springs, 2008.

Sabine Buchholz and Erwin Marsi. CoNLL-X Shared Task on Multilingual Dependency Parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City, June 2006. Association for Computational Linguistics.

Miriam Butt, Helge Dyvik, Tracy Holloway King, Hiroshi Masuichi, and Christian Rohrer. The Parallel Grammar Project. In *COLING-02: Grammar Engineering and Evaluation.* Association for Computational Linguistics, 2002.

Khyathi Chandu, Ekaterina Loginova, Vishal Gupta, Josef van Genabith, Günter Neumann, Manoj Chinnakotla, Eric Nyberg, and Alan W. Black. Code-Mixed Question Answering Challenge: Crowd-sourcing Data and Techniques. In *Proceedings of the Third Workshop* *on Computational Approaches to Linguistic Code-Switching*, pages 29–38, Melbourne, Australia, July 2018. Association for Computational Linguistics.

Angel Chang, Manolis Savva, and Christopher Manning. Interactive Learning of Spatial Knowledge for Text to 3D Scene Generation. In *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*, pages 14–21, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics.

Angel Chang, Manolis Savva, and Christopher D. Manning. Learning Spatial Knowledge for Text to 3D Scene Generation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2028–2038, Doha, Qatar, October 2014. Association for Computational Linguistics.

Angel Xuan Chang. *Text to 3D Scene Generation.* Ph.D. Thesis, Stanford University, December 2015.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C. Lawrence Zitnick. Microsoft COCO Captions: Data Collection and Evaluation Server. *arXiv:1504.00325 [cs]*, April 2015.

S. R. Clay and J. Wilhelms. Put: Language-based interactive manipulation of objects. *IEEE Computer Graphics and Applications*, 16(2):31–39, March 1996.

Bob Coyne and Richard Sproat. WordsEye: An Automatic Text-to-scene Conversion System. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 487–496, New York, NY, USA, 2001. ACM.

Bob Coyne, Daniel Bauer, and Owen Rambow. VigNet: Grounding Language in Graphics using Frame Semantics. In *Proceedings of the ACL 2011 Workshop on Relational Models of Semantics*, pages 28–36, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

Bob Coyne, Cecilia Schudel, Michael Bitz, and Julia Hirschberg. Evaluating a Text-to-Scene Generation System as an Aid to Literacy. In *ISCA International Workshop on Speech* *and Language Technology in Education (SLaTE 2011)*, pages 13–16, Venice, Italy, August 2011.

Robert Coyne. *Painting Pictures with Words - From Theory to System.* Ph.D. Thesis, Columbia University, 2017.

Dick Crouch, Mary Dalrymple, Ron Kaplan, Tracy King, John Maxwell, and Paula Newman. XLE Documentation. `http://ling.uni-konstanz.de/pages/xle/doc/xle_toc.html`, 2011.

Terry Crowley and Nick Thieberger. *Field Linguistics: A Beginner's Guide.* Oxford University Press USA - OSO, Oxford, UNITED KINGDOM, January 2007.

Mary Dalrymple, John Lamping, and Vijay Saraswat. LFG Semantics via Constraints. In *Sixth Conference of the European Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics, 1993.

Mary Dalrymple. *Lexical Functional Grammar*, volume 34 of *Syntax and Semantics.* BRILL, August 2001.

Mark Dras, François Lareau, Benjamin Börschinger, Robert Dale, Yasaman Motazedi, Owen Rambow, Myfany Turpin, and Morgan Ulinski. Complex Predicates in Arrernte. In *Proceedings of the LFG12 Conference.* CSLI Publications, 2012.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. A Neural Network Model for Low-Resource Universal Dependency Parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 339–348, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

Sylvain Dupuy, Arjan Egges, Vincent Legendre, and Pierre Nugues. Generating A 3D Simulation Of A Car Accident From A Written Description In Natural Language: The CarSim System. In *Proceedings of the ACL 2001 Workshop on Temporal and Spatial Information Processing.* Association for Computational Linguistics, 2001.

Paul Ekman. Basic Emotions. In *Handbook of Cognition and Emotion*, pages 45–60. Wiley-Blackwell, 1999.

Endangered Language Alliance. `http://elalliance.org/`.

The Endangered Languages Project. `http://www.endangeredlanguages.com/`.

A. Esuli and F. Sebastiani. SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy, 2006. European Language Resources Association (ELRA).

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9(Aug):1871–1874, 2008.

Michele I. Feist and Dedre Gentner. On Plates, Bowls, and Dishes: Factors in the Use of English IN and ON. In *Proceedings of the Twentieth Annual Meeting of the Cognitive Science Society*, pages 345–349, Hillsdale, NJ, 1998. Erlbaum.

Christiane Fellbaum and George Miller. *WordNet: An Electronic Lexical Database.* MIT Press, 1998.

Charles J. Fillmore, Christopher R. Johnson, and Miriam R. L. Petruck. Background to Framenet. *International Journal of Lexicography*, 16(3):235–250, September 2003.

James J. Gibson. The Theory of Affordances. In *The Ecological Approach to Visual Perception.* Erlbaum, 1977.

Jost Gippert, Nikolaus P. Himmelmann, and Ulrike Mosel. *Essentials of Language Documentation.* Number 178 in Trends in Linguistics. De Gruyter, Inc., Berlin/Boston, GERMANY, 2006.

Kevin Glass. *Automating the Conversion of Natural Language Fiction to Multi-Modal 3D Animated Virtual Environments.* Ph.D. Thesis, Rhodes University, Grahamstown, South Africa, August 2008.

Yoav Goldberg and Michael Elhadad. An Efficient Algorithm for Easy-First Non-Directional Dependency Parsing. In *Human Language Technologies: The 2010 Annual Conference* *of the North American Chapter of the Association for Computational Linguistics*, pages 742–750, Los Angeles, California, June 2010. Association for Computational Linguistics.

Jenny Green. *A Learner's Guide to Eastern and Central Arrernte.* IAD Press, Alice Springs, Australia, 1994.

Lenore A. Grenoble and N. Louanna Furbee. *Language Documentation: Practice and Values.* John Benjamins Publishing Company, Amsterdam, NETHERLANDS, 2010.

Colette Grinevald. Endangered Languages of Mexico and Central America. In *Language Diversity Endangered*, pages 59–86. De Gruyter Mouton, Berlin, Boston, 2008.

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. Cross-lingual Dependency Parsing Based on Distributed Representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1234–1244, Beijing, China, July 2015. Association for Computational Linguistics.

Birgit Hamp and Helmut Feldweg. GermaNet - a Lexical-Semantic Net for German. In *Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications.* Association for Computational Linguistics, 1997.

Kaveh Hassani and Won-Sook Lee. Visualizing Natural Language Descriptions: A Survey. *ACM Computing Surveys (CSUR)*, 49(1):17:1–17:34, June 2016.

John Henderson and Veronica Dobson. *Eastern and Central Arrernte to English Dictionary.* IAD Press, Alice Springs, 1994.

John Henderson. *Topics in Eastern and Central Arrernte Grammar.* PhD Thesis, University of Western Australia, 1998.

Verena Henrich and Erhard Hinrichs. GernEdiT - The GermaNet Editing Tool. In *Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, 2010. European Languages Resources Association (ELRA).

Annette Herskovits. *Language and Spatial Cognition: An Interdisciplinary Study of the Prepositions in English.* Cambridge University Press, December 1986.

Jane H. Hill and Kenneth C. Hill. *Speaking Mexicano: Dynamics of Syncretic Language in Central Mexico.* University of Arizona Press, 1986.

M. Hodosh, P. Young, and J. Hockenmaier. Framing Image Description as a Ranking Task: Data, Models and Evaluation Metrics. *Journal of Artificial Intelligence Research*, 47:853–899, August 2013.

P. Jain, H. Darbari, and V. C. Bhavsar. Spatial intelligence from hindi language text for scene generation. In *2017 2nd International Conference for Convergence in Technology (I2CT)*, pages 132–138, April 2017.

Ronald M. Kaplan and Joan Bresnan. Lexical-Functional Grammar: A Formal System for Grammatical Representation. In *Formal Issues in Lexical-Functional Grammar.* Center for the Study of Language (CSLI), November 1995.

Lauri Karttunen, Tamás Gaál, and André Kempe. Xerox finite-state tool, 1997.

Frances E. Karttunen. *An Analytical Dictionary of Nahuatl.* University of Oklahoma Press, 1992.

Yılmaz Kılıçaslan, Özlem Uçar, and Edip Serdar Güner. An NLP-Based 3D Scene Generation System for Children with Autism or Mental Retardation. In Leszek Rutkowski, Ryszard Tadeusiewicz, Lotfi A. Zadeh, and Jacek M. Zurada, editors, *Artificial Intelligence and Soft Computing – ICAISC 2008*, Lecture Notes in Computer Science, pages 929–938. Springer Berlin Heidelberg, 2008.

Ophélie Lacroix, Lauriane Aufrant, Guillaume Wisniewski, and François Yvon. Frustratingly Easy Cross-Lingual Transfer for Transition-Based Dependency Parsing. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1058–1063, San Diego, California, June 2016. Association for Computational Linguistics.

Francois Lareau, Mark Dras, Benjamin Borschinger, and Robert Dale. Collocations in Multilingual Natural Language Generation: Lexical Functions meet Lexical Functional Grammar. In *Proceedings of the Australasian Language Technology Association Workshop 2011*, pages 95–104, Canberra, Australia, December 2011.

François Lareau, Mark Dras, Benjamin Börschinger, and Myfany Turpin. Implementing lexical functions in XLE. In *Proceedings of LFG12*, pages 362–382, Bali, 2012.

François Lareau. Arrernte Footy: A natural language generation project in Arrernte. `http://web.science.mq.edu.au/~ayeye/`, 2012.

Stephen C. Levinson. Motion verb stimulus, version 2. In *Manual for the Field Season 2001*, pages 9–13. Max Planck Institute for Psycholinguistics, Nijmegen, 2001.

Stephen C. Levinson. *Space in Language and Cognition: Explorations in Cognitive Diversity.* Cambridge University Press, Cambridge, UK, 2003.

Minhua Ma. *Automatic Conversion of Natural Language to 3D Animation.* Ph.D. Thesis, University of Ulster, 2006.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

Andre Martins, Noah Smith, Eric Xing, Pedro Aguiar, and Mario Figueiredo. Turbo Parsers: Dependency Parsing by Approximate Variational Inference. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 34–44, Cambridge, MA, October 2010. Association for Computational Linguistics.

Mike Maxwell and Jonathan D. Amith. Language Documentation: The Nahuatl Grammar. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, Lecture Notes in Computer Science, pages 474–485. Springer Berlin Heidelberg, 2005.

Stephen McConnel and H. Andrew Black. PC-PATR Reference Manual. `http://software.sil.org/downloads/r/pc-patr/pcpatr.html`, November 2006.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajic. Non-Projective Dependency Parsing using Spanning Tree Algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada, October 2005. Association for Computational Linguistics.

Ryan McDonald, Kevin Lerman, and Fernando Pereira. Multilingual Dependency Analysis with a Two-Stage Discriminative Parser. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 216–220, New York City, June 2006. Association for Computational Linguistics.

George A. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41, November 1995.

Joakim Nivre, Johan Hall, and Jens Nilsson. MaltParser: A Data-Driven Parser-Generator for Dependency Parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy, 2006. European Language Resources Association (ELRA).

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. The CoNLL 2007 Shared Task on Dependency Parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135, June 2007.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. Universal Dependencies v1: A Multilingual Treebank

Collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 1659–1666, Paris, France, May 2016.

Rachel Nordlinger and Joan Bresnan. Lexical-Functional Grammar: Interactions Between Morphology and Syntax. In *Non-Transformational Syntax*, pages 112–140. Wiley-Blackwell, 2011.

Rachel Nordlinger. *Constructive Case: Dependent-Marking Nonconfigurationality in Australia.* Ph.D., Stanford University, 1997.

Donald A Norman. *The Psychology of Everyday Things.* Basic Books, New York, 1988. OCLC: 874159470.

Maghnus O'Kane, Joe Carthy, and Michela Bertolotto. Text-to-scene Conversion for Accident Visualization. In *ACM SIGGRAPH 2004 Posters*, SIGGRAPH '04, page 63, Los Angeles, California, 2004.

Petr Pajas and Jan Štěpánek. Recent Advances in a Feature-Rich Framework for Treebank Annotation. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 673–680, Manchester, UK, August 2008. COLING 2008 Organizing Committee.

ParGram / ParSem: An international collaboration on LFG-based grammar and semantics development. `https://pargram.w.uib.no/`, 2013.

S. Parisi, J. Bauch, J. Berssenbrugge, and R. Radkowski. Ontology-driven Generation of 3D Animations for Training and Maintenance. In *2007 International Conference on Multimedia and Ubiquitous Engineering (MUE'07)*, pages 608–614, April 2007.

Hanchuan Peng, Fuhui Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, August 2005.

Miriam R L Petruck and Michael J Ellsworth. Representing Spatial Relations in FrameNet. In *Proceedings of the First International Workshop on Spatial Language Understanding*, pages 41–45, New Orleans, June 2018. Association for Computational Linguistics.

Dimitri Plemenos and Georgios Miaoulis. Intelligent scene modeling. In Dimitri Plemenos and Georgios Miaoulis, editors, *Visual Complexity and Intelligent Computer Graphics Techniques Enhancements*, Studies in Computational Intelligence, pages 27–64. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

Princeton University. WordNet: A Lexical Database for English. `https://wordnet.princeton.edu/`.

James Pustejovsky, Parisa Kordjamshidi, Marie-Francine Moens, Aaron Levine, Seth Dworman, and Zachary Yocum. SemEval-2015 Task 8: SpaceEval. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 884–894, Denver, Colorado, June 2015. Association for Computational Linguistics.

James Pustejovsky. ISO-Space: Annotating Static and Dynamic Spatial Information. In Nancy Ide and James Pustejovsky, editors, *Handbook of Linguistic Annotation*, pages 989–1024. Springer Netherlands, Dordrecht, 2017.

Cyrus Rashtchian, Peter Young, Micah Hodosh, and Julia Hockenmaier. Collecting Image Annotations Using Amazon's Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 139–147, Los Angeles, June 2010. Association for Computational Linguistics.

Kellie Rolstad. Language Death in Central Mexico: The Decline of Nahuatl and the New Bilingual Maintenance Programs. *Bilingual Review / La Revista Bilingüe*, 26(1):3–18, 2001.

Rick Rosenberg. Tools for Activating Materials and Tasks in the English Language Classroom. *English Teaching Forum*, 47(4):2, 2009.

Josef Ruppenhofer, Michael Ellsworth, Miriam R. L Petruck, Christopher R. Johnson, Collin F. Baker, and Jan Scheffczyk. FrameNet II: Extended Theory and Practice. `https://framenet2.icsi.berkeley.edu/docs/r1.7/book.pdf`, November 2016.

Susan Rvachew and Françoise Brosseau-Lapré. *Developmental Phonological Disorders : Foundations of Clinical Practice.* Plural Publishing, Inc, San Diego, 2012.

Melanie Siegel. Open German WordNet. `https://github.com/hdaSprachtechnologie/odenet`.

SIL International. FieldWorks. `https://software.sil.org/fieldworks/`.

Robert F. Simmons. The Clowns Microworld. In *Theoretical Issues in Natural Language Processing*. Association for Computational Linguistics, 1975.

Jane Simpson. Expressing pragmatic constraints on word order in Warlpiri. In Annie Zaenen, Jane Simpson, Tracy Holloway King, Jane Grimshaw, and Joan Maling, editors, *Architectures, Rules, and Preferences; Variations on Themes by Joan W. Bresnan*, pages 403–427. CSLI Publications, 2007.

T. G. H. Strehlow. *Aranda Phonetics and Grammar*. Number 7 in Oceania Monographs. Australian National Research Council, Sydney, 1944.

Yla R. Tausczik and James W. Pennebaker. The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods. *Journal of Language and Social Psychology*, 29(1):24–54, March 2010.

Morgan Ulinski, Victor Soto, and Julia Hirschberg. Finding Emotion in Image Descriptions. In *Proceedings of the First International Workshop on Issues of Sentiment Discovery and Opinion Mining*, WISDOM '12, pages 8:1–8:7, New York, NY, USA, 2012. ACM.

Morgan Ulinski, Anusha Balakrishnan, Daniel Bauer, Bob Coyne, Julia Hirschberg, and Owen Rambow. Documenting Endangered Languages with the WordsEye Linguistics Tool. In *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 6–14, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics.

Morgan Ulinski, Anusha Balakrishnan, Bob Coyne, Julia Hirschberg, and Owen Rambow. WELT: Using Graphics Generation in Linguistic Fieldwork. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 49–54, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

Morgan Ulinski, Julia Hirschberg, and Owen Rambow. Incrementally Learning a Dependency Parser to Support Language Documentation in Field Linguistics. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 440–449, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee.

Morgan Ulinski, Julia Hirschberg, and Owen Rambow. Multilingual Spatial Relation and Motion Treebank. `https://doi.org/10.7916/D8W959HJ`, 2016.

Morgan Ulinski, Bob Coyne, and Julia Hirschberg. Evaluating the WordsEye Text-to-Scene System: Imaginative and Realistic Sentences. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, pages 1493–1499, Miyazaki, Japan, 2018. European Language Resource Association.

Morgan Ulinski, Bob Coyne, and Julia Hirschberg. SpatialNet: A Declarative Resource for Spatial Relations. In *Proceedings of the Combined Workshop on Spatial Language Understanding (SpLU) and Grounded Communication for Robotics (RoboNLP)*, pages 61–70, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

Universal Dependencies. `https://universaldependencies.org/`.

Dmitry Ustalov. A text-to-picture system for russian language. *RuSSIR 2012*, 2012.

Piek Vossen, editor. *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*. Springer Netherlands, 1998.

W. Y. Wang, D. Bohus, E. Kamar, and E. Horvitz. Crowdsourcing the acquisition of natural language corpora: Methods and observations. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 73–78, December 2012.

Daniel A. Werning. BowPed TRPS Data, English_4. `https://doi.org/10.17171/2-5-2`, 2016.

CYNTHIA M. Whissell. The Dictionary of Affect in Language. In Robert Plutchik and Henry Kellerman, editors, *The Measurement of Emotions*, pages 113–131. Academic Press, January 1989.

David P. Wilkins. *Mparntwe Arrernte (Aranda): Studies in the Structure and Semantics of Grammar*. PhD Thesis, The Australian National University, 1989.

Stephanie Wood. Online Nahuatl Dictionary. `https://nahuatl.uoregon.edu/`.

XLE Project. `http://ling.uni-konstanz.de/pages/xle/`.

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

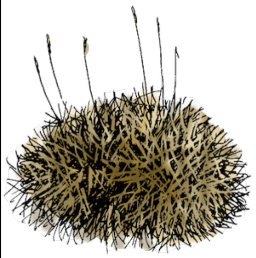
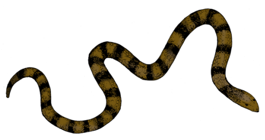
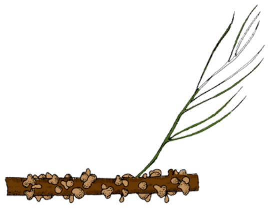
Omar F. Zaidan and Chris Callison-Burch. Crowdsourcing Translation: Professional Quality from Non-Professionals. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1220–1229, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

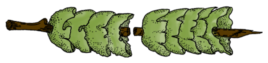
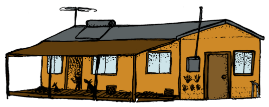
Xin Zeng, Q. H. Mehdi, and N. E. Gough. From visual semantic parameterization to graphic visualization. In *Ninth International Conference on Information Visualisation (IV'05)*, pages 488–493, July 2005.

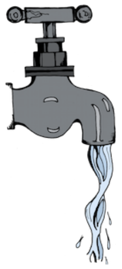
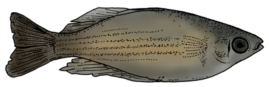
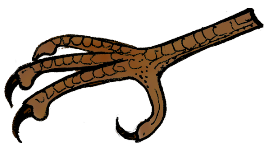
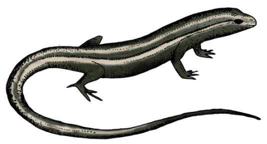
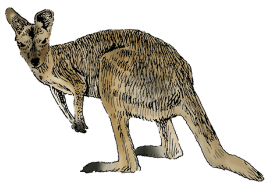
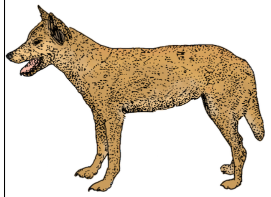
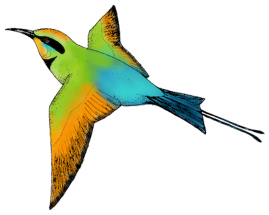
Appendix A

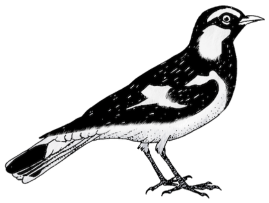
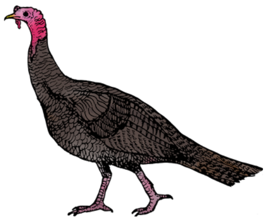
Custom Arrernte-Related WordsEye Content

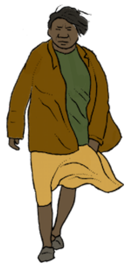
The table below shows some of the custom content we created to adapt WordsEye for Arrernte culture, as described in Chapter 3, Section 3.6.1.1. The images in this appendix are based on pictures from IAD Press, used with permission, which we enhanced and cropped in PhotoShop.

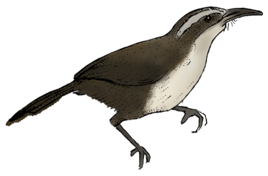
 *aywerte* 'spinifex'	 *arlkenye-arlkenye, untewarrerte* 'stripy (snake)'	 *arlperle* 'honey dew'	*ingkwerlpe* 'rock pituri'

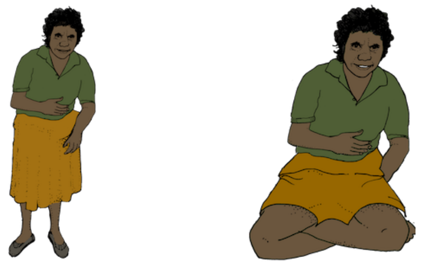
ahernenge 'grub from red river gum'	*untyeme alheme* 'go back from or face away from'	*thelelheme, altyiwelheme* 'bucket'	*ingkele atweme* 'kick'
unteme 'run'	*arlpelhe* 'wing, feather'	 *inte, interlpe* 'skewer'	*arlatyeye* 'pencil yam'
pirtwerre, pirterre atnwengke 'plain pituri'	*mpwaltye* 'type of a frog'	*atwakeye* 'wild orange'	 *Warle* 'house'

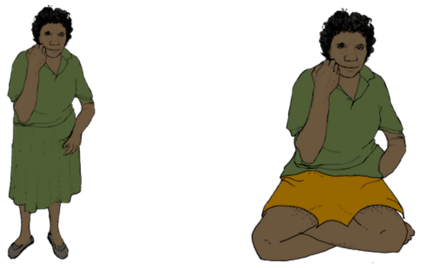
 *kwatye* ‘water’	 *irrpennge, artepinye* ‘fish’	 *ingkelthele* ‘claw’	*yerrampe* ‘honeyant’
kwarte ‘egg’	 *ikwarre* ‘type of skink’	*aherre* ‘kangaroo’	 *arenge, apwerte-arenye* ‘euro, wallaroo’
 *akngwelye artnwere* ‘dingo’	*kwarlpe* ‘hare wallaby’	*atnethwekethweke* ‘button quail’	 *tirre-tirre* ‘rainbow bee-eater’

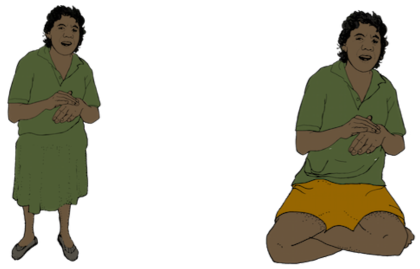
apelkere 'crested pigeon'	*kwepalepale* 'crested bellbird'	 *teye-teye, rteye-rteye* 'magpie-lark'	*irrkerlantye* 'black kite'
pmwilyare 'bush thick-knee'	*atnarre-tharnke-tharnke* 'black-tailed native hen'	*atyankerne* 'type of mistletoe'	 *karpele-karpele* 'turkey'

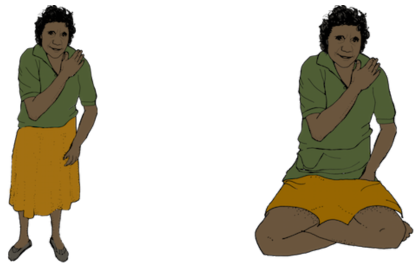

ularre apeyteme ‘come or face towards’

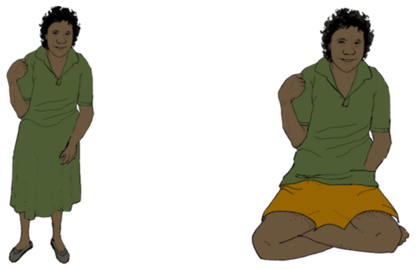

atyewaketye ‘grey-crowned babbler’

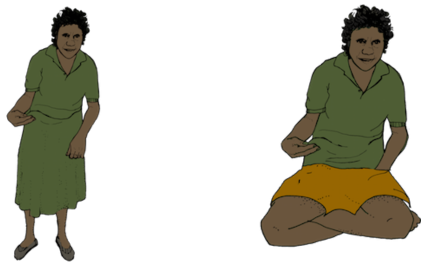

[hand gesture] *ampe* ‘child’

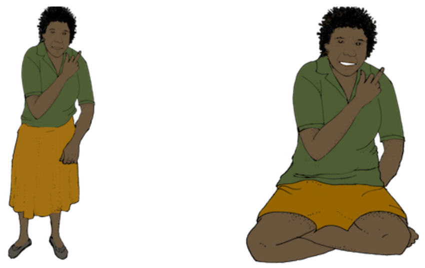
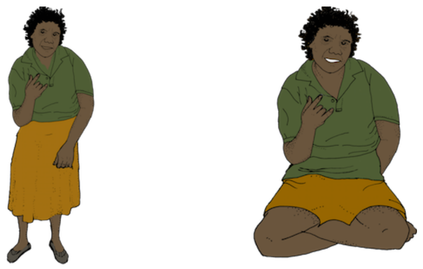
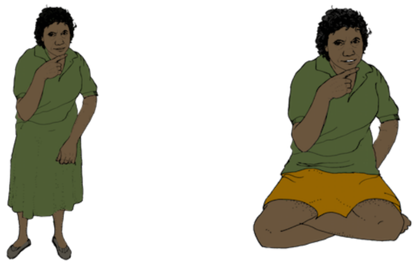
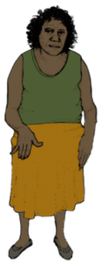

[hand gesture] *atyeye* ‘younger sibling’


[hand gesture] *arperle* ‘grandparent (father’s mother and her brothers and sisters)’


[hand gesture] *ipmenhe* ‘grandparent (grandmother, mother’s mother and her brothers and sisters’


[hand gesture] *atyeye* ‘younger sibling’


[hand gesture] *kake, akngerrepate* ‘elder brother’

 [hand gesture] *arrenge* ‘grandparent (father’s father and his brothers and sisters)’	 [hand gesture] *meye* ‘mother’
 [hand gesture] *akngeye* ‘father’	 [hand gesture] *atyemeye* ‘grandparent (mother’s father and his brothers and sisters)’

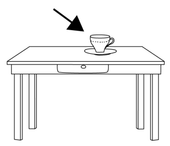
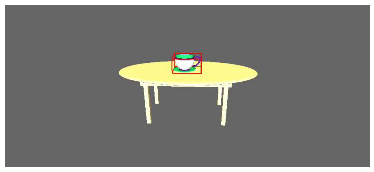
Appendix B

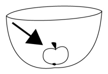
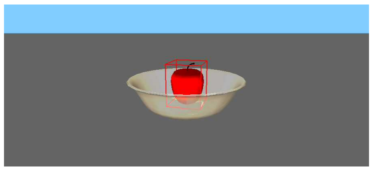
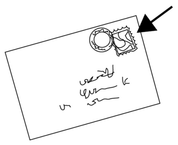
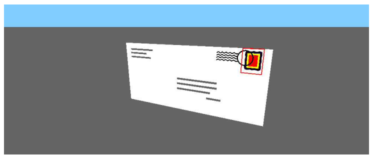
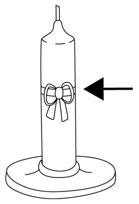
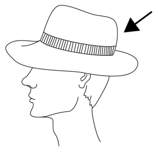
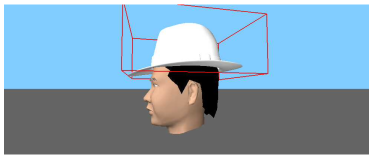
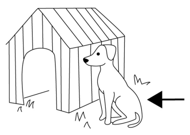
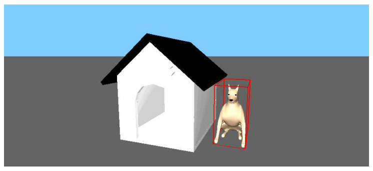
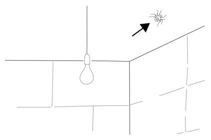
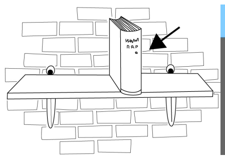
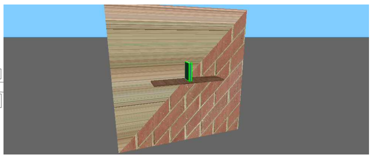
Nahuatl Topological Relations

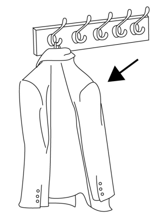
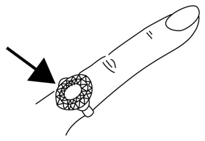
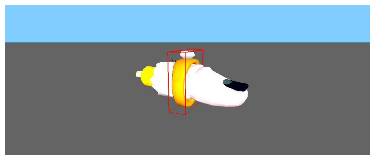
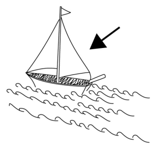
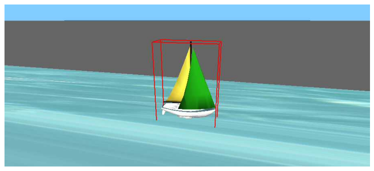
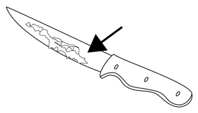
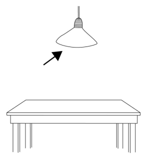
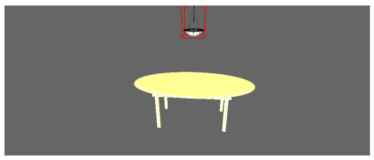
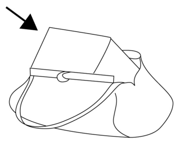
This appendix shows the WordsEye scenes created based on the Max Planck topological relations picture series (Section B.1) and Nahuatl descriptions we elicited for these scenes (Section B.2).

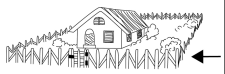
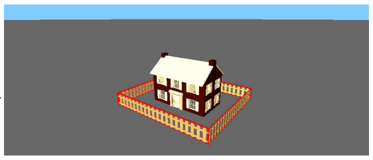
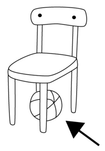
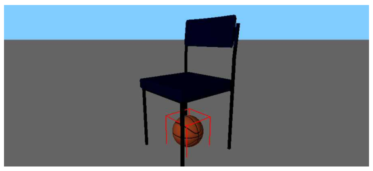
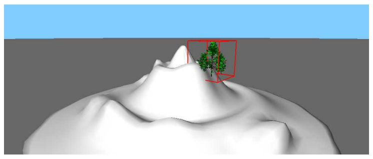
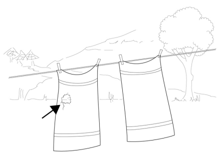
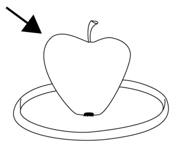
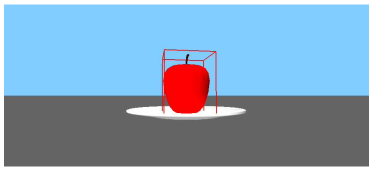
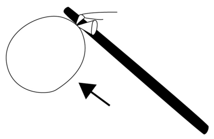
B.1 WordsEye scenes for Max Planck Topological Relations

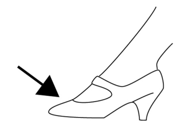
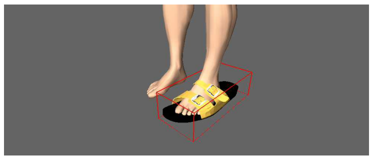
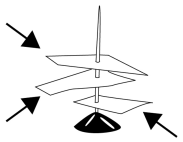
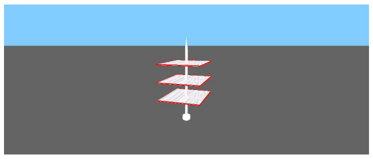
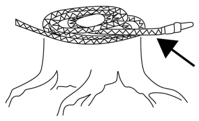
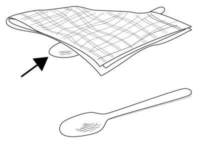
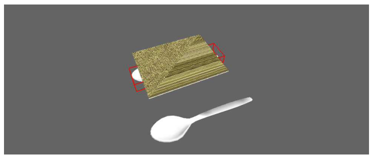
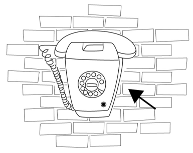
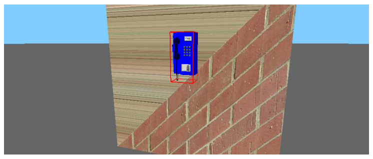
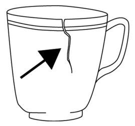
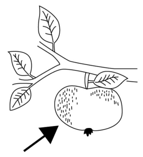
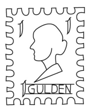
The following table shows the WordsEye scenes created for each picture in the Max Planck topological relations picture series, along with the input text that was used to generate the scene. Pictures we were unable to adequately duplicate in WordsEye are marked as N/A. For the pictures we were unable to duplicate in WordsEye, we also note the problems that prevented us from creating an appropriate scene.

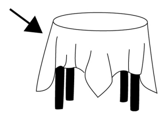
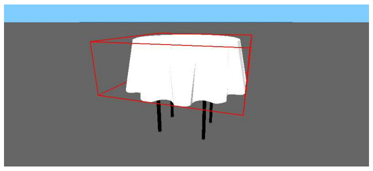
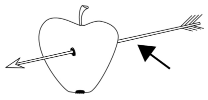
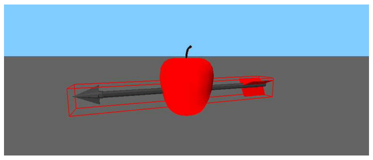
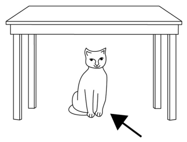
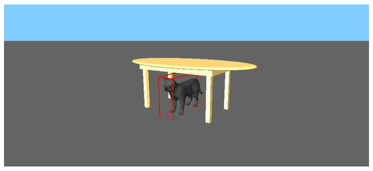
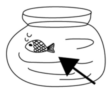
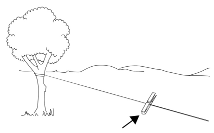
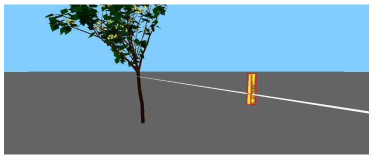
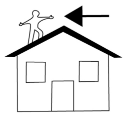
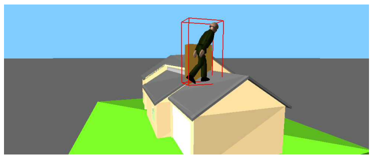
	Max Planck Picture	WordsEye Scene	Input Text
1.			The huge teacup is on the dining room table.

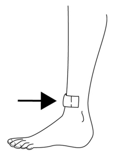
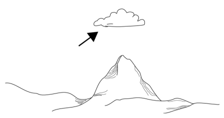
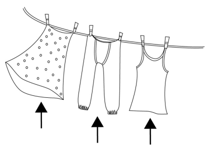
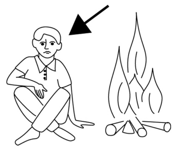
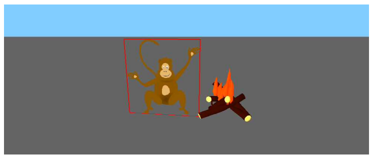
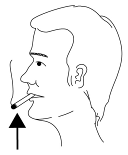
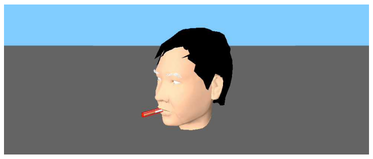
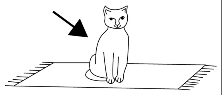
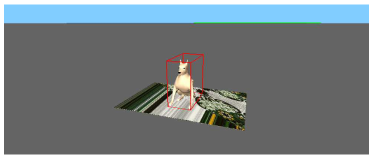
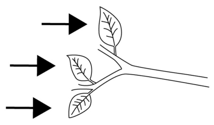
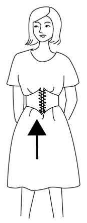
	Max Planck Picture	WordsEye Scene	Input Text
2.			The small apple is in the transparent cereal bowl.
3.			the envelope. the invisible red block is -.05 inches in front of the envelope. The block is 1 inch tall and .8 inches wide and .05 inches deep. The block is -1 inches above the envelope. The block is -1.1 inches to the right of the envelope.
4.		N/A. Missing 3D objects: candle, ribbon. No graphical support for flexible object like ribbon or "fit around" relation.	
5.			The head. The fedora is -4.4 inches above the head. The fedora is 13.8 inches wide. It is -12.25 inches behind the head.
6.			The small dog is on the right of the white doghouse.
7.		N/A. Missing 3D object: spider.	
8.			The shelf is on the brick wall. The book is on the shelf. The book is facing right. It is 2.5 inches in front of the wall.

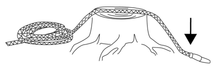
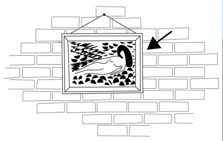
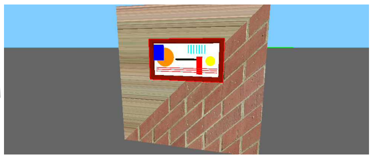
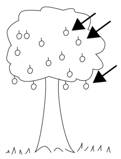
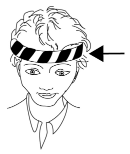
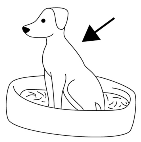
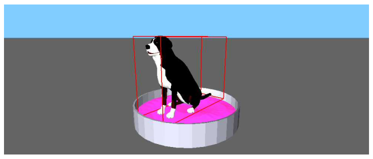
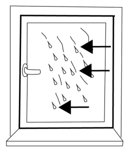
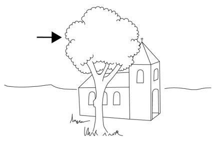
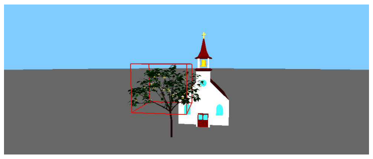
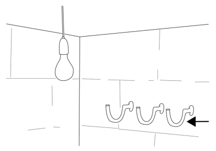
	Max Planck Picture	WordsEye Scene	Input Text
9.		N/A. Missing 3D objects: hooks, coat.	
10.			The ring is in the pink finger. the ring is 1.7 inches tall. The ring is -1.5 inches above the finger.
11.			The sailboat is in the water. The sailboat is -12 inches above the water.
12.		N/A. No graphical support for "butter" consistency on knife.	
13.			The light is 3 feet above the dining room table.
14.		N/A. Missing 3D object: open bag.	

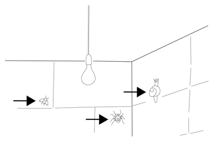
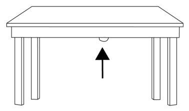
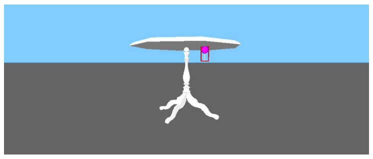
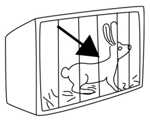
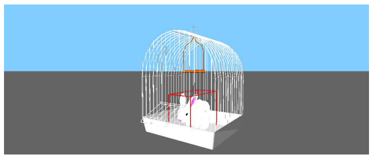
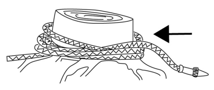
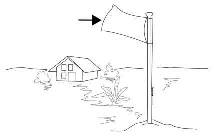
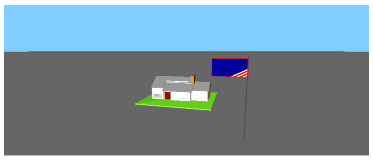
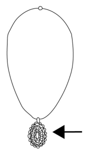
Max Planck Picture	WordsEye Scene	Input Text
15. 		The small house is 6 feet behind the first fence. The first fence is 25 feet wide. It is 3 feet tall. The second fence is 5 feet to the right of the house. It is 22 feet wide and 3 feet tall. It is facing left. It is -16 feet behind the house. The third fence is 6 feet behind the house. It is 25 feet wide and 3 feet tall. The fourth fence is 5 feet to the left of the house. It is 22 feet long and 3 feet tall. It is facing right. It is -16 feet behind the house.
16. 		The basketball is under the chair.
17.		The gray circle is to the right of the grey mountain. the 12 foot tall tree is 31.5 feet left of the circle. the tree is 16 feet above the ground.
18. 	N/A. No graphical support for putting a hole in an object.	
19. 		The apple is on the plate.
20. 	N/A. Missing 3D object: balloon. No graphical support for tying an object on a stick.	

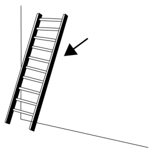
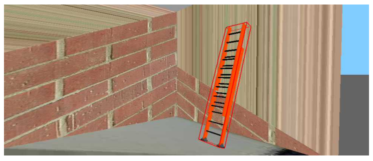
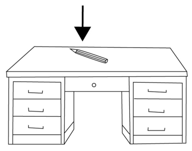
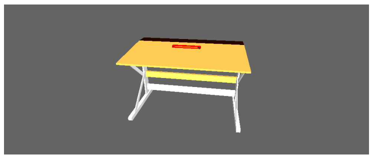
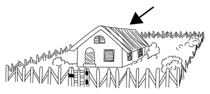
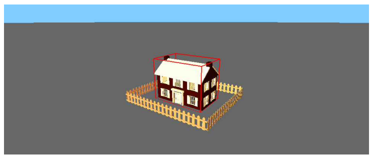
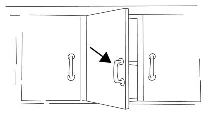
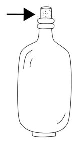
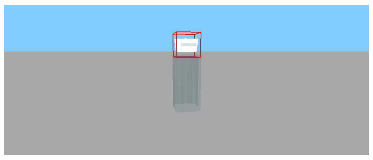
	Max Planck Picture	WordsEye Scene	Input Text
21.			The swimmer is -4.2 inches above the shoe. The shoe is 7.2 inches wide. The swimmer is -34 inches to the left of the shoe.
22.			The upside down nail. The nail is 10 inches tall and 1 inches wide and 1 inches deep. the small yellow paper is -3 inches above the nail. the small green paper is -5 inches above the nail. The red small paper is -7 inches above the nail.
23.		N/A. Missing 3D object: coiled rope.	
24.			The small handkerchief is on the first large spoon. The second large spoon is 6 inches to the left of the first spoon.
25.			The telephone is in front of the brick wall. It is 4 feet above the ground.
26.		N/A. No graphical support for putting a crack in an object; no 3D object of a cracked cup.	
27.		N/A. No 3D object for a branch with leaves.	
28.		N/A. No 3D object of a stamp with a face on it.	

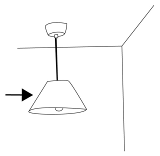
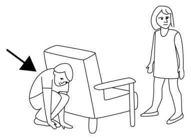
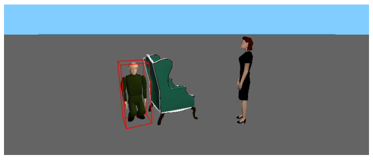
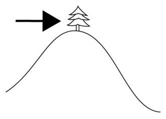
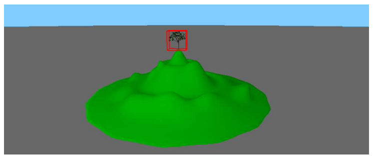
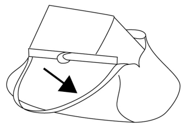
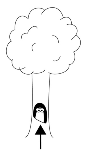
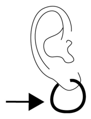
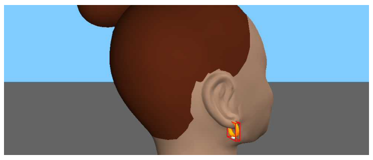
	Max Planck Picture	WordsEye Scene	Input Text
29.			the second table is three feet high. It is eight feet wide. It is eight feet deep. The second table is in the black dining room table. The dining room table is six feet wide. The dining room table is four feet tall.
30.			The small arrow is in the apple. It is -5 inches above the apple.
31.			The big house cat is under the dining room table.
32.		N/A. No 3D object of a fish bowl	
33.			The cylinder is supine. It is 20 feet long and .3 inches wide and .3 inches deep. The tree is -5.5 feet to the left of the cylinder. It is 12 feet tall. It is on the ground. It is -5.5 feet behind the cylinder. The cylinder is 4 feet above the ground. The large clothespin is -2.5 inches above the cylinder. It is upside down. It is -7 feet to the right of the cylinder. It is facing right.
34.			The soldier is -12.5 feet in front of the house. The soldier is -3.8 feet above the house. The soldier is -7 feet to the right of the house. The soldier is facing north. The soldier is running.

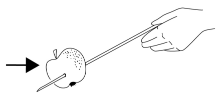
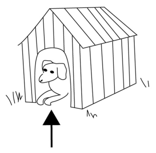
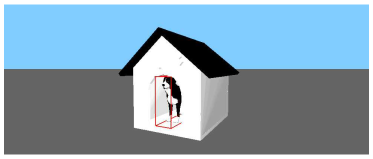
Max Planck Picture		WordsEye Scene	Input Text
35.		N/A. No 3D object for a band-aid	
36.		N/A. No 3D object of a cloud.	
37.		N/A. No 3D objects for shirt or dress.	
38.			The monkey is next to the fire.
39.			The cigarette is -2 inches in front of the head. The cigarette is supine. The cigarette is 3 inches above the ground.
40.			The small dog is on the large placemat.
41.		N/A. No 3D object for a branch with leaves on it.	
42.		N/A. No graphical support for flexible objects like belts. No 3D object for a person with a belt.	

	Max Planck Picture	WordsEye Scene	Input Text
43.		N/A. No graphical support for conforming the shape of a rope to the stump.	
44.			The painting is in front of the brick wall. The painting is 4 feet above the ground.
45.		N/A. No fruit tree object. No support for laying out collections of objects in anything but a line.	
46.		N/A. No 3D object for a head with a headband. No graphical support for a "fit on" or "fit around" spatial primitive.	
47.			The dog is -.4 inches above the petri dish. the dog is 4 inches tall.
48.		N/A. Missing graphical object: raindrops.	
49.			The church. The big tree is 8 feet in front of the church. It is -18 feet to the left of the church. The invisible red cube is 25 feet wide and 22 feet tall and 25 feet deep. It is -22 feet above the tree.
50.		N/A. Missing graphical objects: hooks.	

	Max Planck Picture	WordsEye Scene	Input Text
51.		N/A. No graphical support for "fit around" spatial primitive. No necklace 3D object.	
52.		N/A. No 3D bug objects.	
53.			the lollipop is in the table. The lollipop is -5.7 inches above the table. It is -1 feet to the right of the table.
54.			The rabbit is -2 feet above the cage. It is -20 inches in front of the cage.
55.		N/A. No graphical support for "fit around" relation.	
56.			The small house. The stick is 5 feet to the right of the house. The stick is 10 feet in front of the house. The stick is 10 feet tall. The flag is -28 inches above the stick. The flag is -4 feet to the right of the stick.
57.		N/A. No 3D object for a necklace or pendant.	

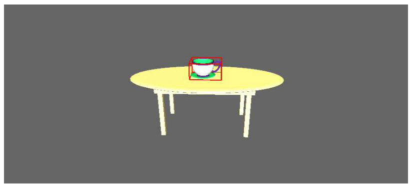
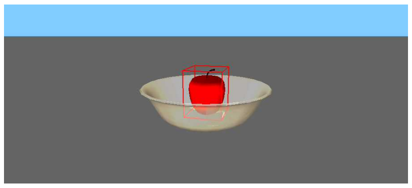
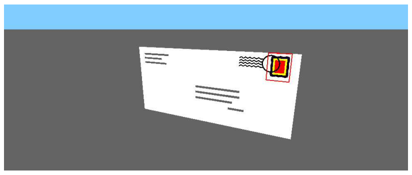
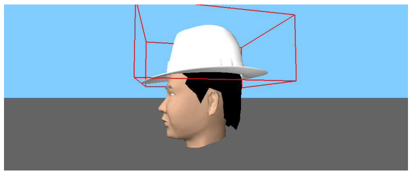
	Max Planck Picture	WordsEye Scene	Input Text
58.			the ceiling is on the first brick wall. the wall is on the floor. it is 20 feet wide. the second brick wall is facing right. it is 20 feet wide. it is to the right of the first brick wall. the extension ladder is to the left of the second brick wall. it is 6 feet tall. it is leaning to the west. it is 5 feet in front of the first brick wall.
59.			The pencil is above the small desk. The pencil is supine.
60.			The small house is 6 feet behind the first fence. The first fence is 25 feet wide. It is 3 feet tall. The second fence is 5 feet to the right of the house. It is 22 feet wide and 3 feet tall. It is facing left. It is -16 feet behind the house. The third fence is 6 feet behind the house. It is 25 feet wide and 3 feet tall. The fourth fence is 5 feet to the left of the house. It is 22 feet long and 3 feet tall. It is facing right. It is -16 feet behind the house.
61.		N/A. No 3D object of a cabinet with an open door.	
62.			The bottle is 1 inch above the ground. The ground is white. The invisible block is 4 inches wide and 4 inches deep and 4 inches tall. It is -3 inches above the bottle.

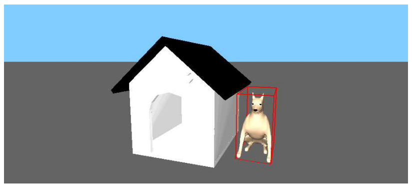
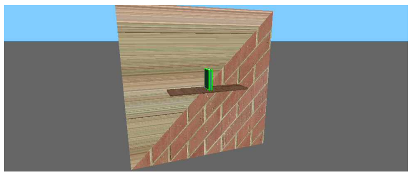
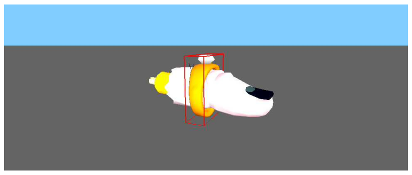
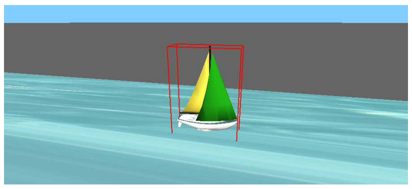
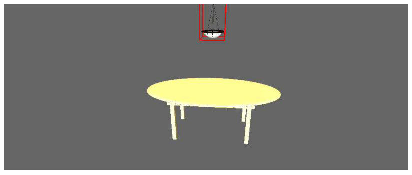
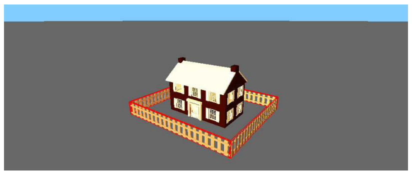
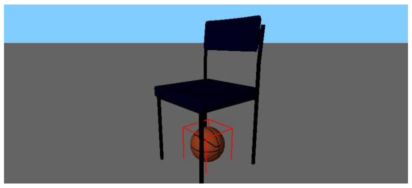
	Max Planck Picture	WordsEye Scene	Input Text
63.		N/A. No graphical support for hanging an object from the ceiling.	
64.			The soldier is 0.1 feet behind the chair. The soldier is crouching. The soldier is 4.5 feet tall. The soldier is facing left. The soldier is -.9 feet above the ground. The woman is 5 feet tall. The woman is facing the chair. The woman is 1.5 feet in front of the chair. The woman is .5 feet to the left of the chair.
65.			The small tree is on the mountain.
66.		N/A. No 3D object of a bag with handles.	
67.		N/A. No 3D object of a tree with a hole in it; no graphical support for putting holes in objects.	
68.			the cube is -1.5 feet above the boy. it is 8 inches wide and 2.5 inches tall and .1 inches deep. it has a “Columbia” texture. it is -1.9 inches in front of the boy.
69.			The head. The small ring is -.7 inch to the left of the head. The ring is upside down. The ring is 2.7 inches above the ground. The ring is -5.5 inches behind the head.

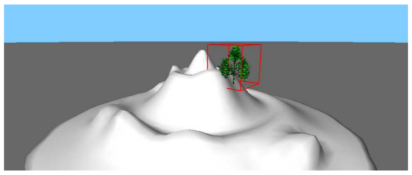
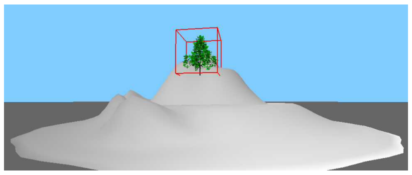
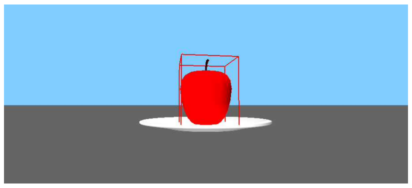
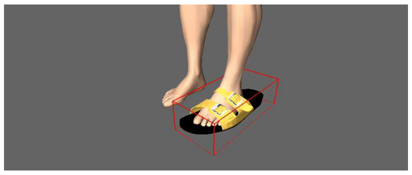
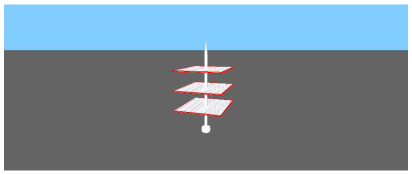
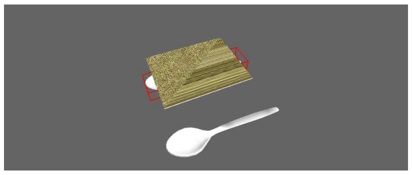
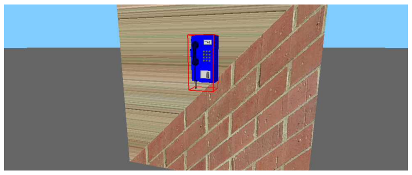
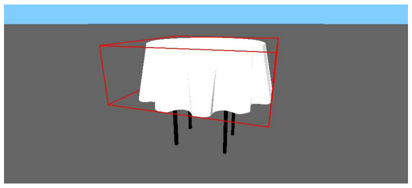
	Max Planck Picture	WordsEye Scene	Input Text
70.		N/A. No graphical support for a "grasp" or "hold" relation.	
71.			The dog is in the white doghouse. The dog is 1.3 feet tall. The dog is -2.5 feet above the doghouse. The dog is -27 inches in front of the doghouse.

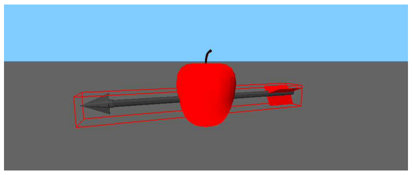
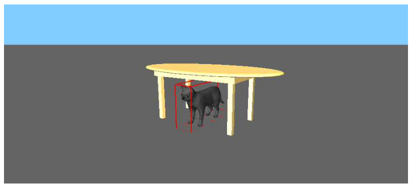
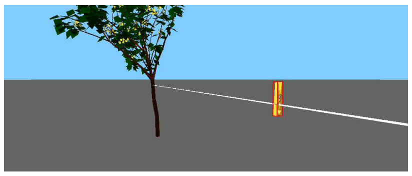
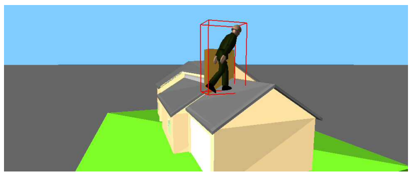
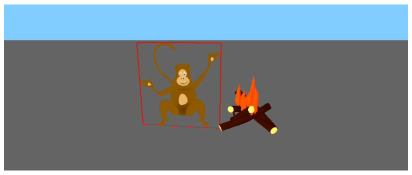
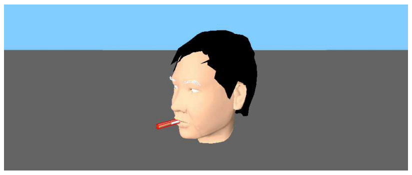
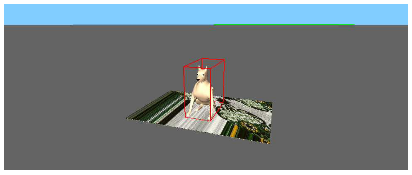
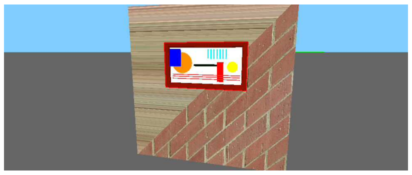
B.2 Nahuatl elicitations

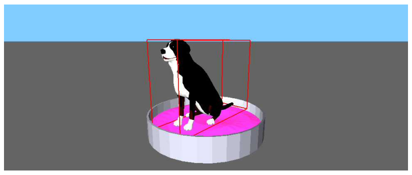
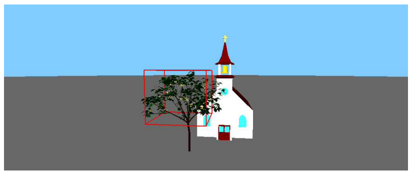
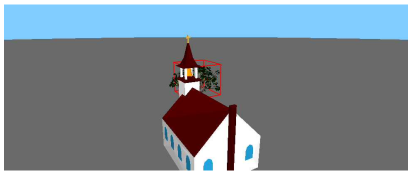
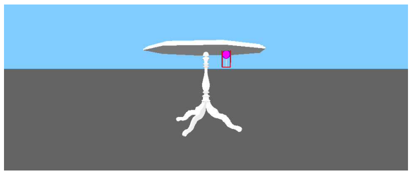
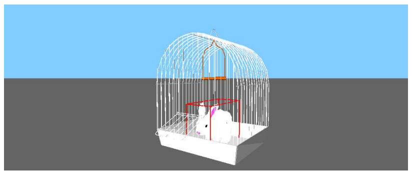
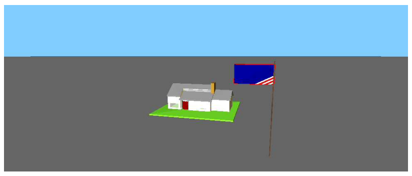
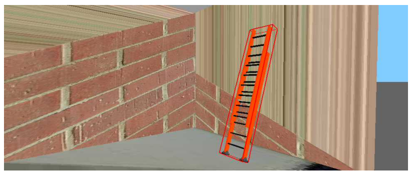
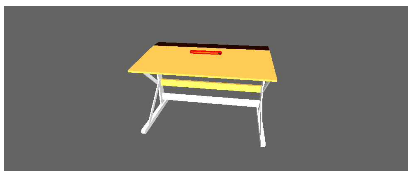
The following table shows the Nahuatl descriptions we elicited for the WordsEye topological relations scenes, including any followup scenes we created during the elicitation session.

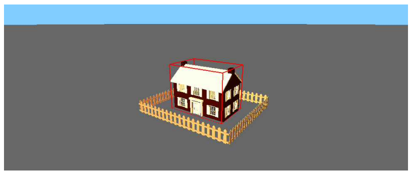
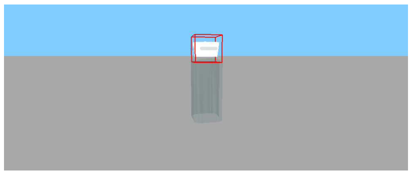
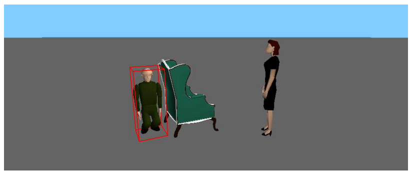
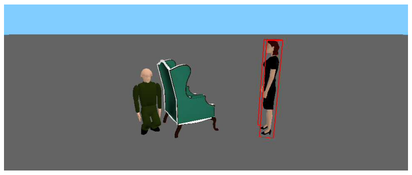
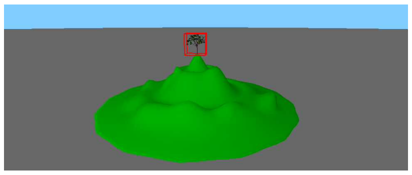
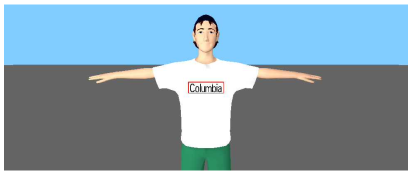
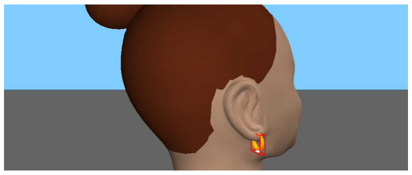
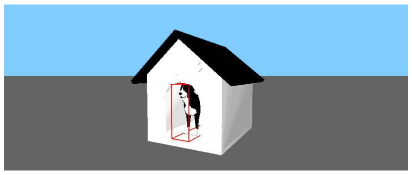
	WordsEye Scene	Nahuatl description and gloss
1.		*in ʃalo ka ipan tłapetʃ* the cup is in table
2.		*mansana ka itetʃ in kakatsi* apple is inside the bowl
3.		*namatsi katsekotok ipan we amatł* little.paper stick in big paper
5.		*intłakatł pia se sombrero* man has a hat

	WordsEye Scene	Nahuatl description and gloss
6.		*in inskwintɬi kaja iwatok inawok itʃa* the dog is sit together house
8.		(a) *in amoʃtɬi ka itok* the book is standing (b) *in amoʃtɬi ka tepamitɬ* the book is wall
10.		*in tepos pia se mapil* the piece.of.metal(ring) has one finger
11.		(a) *in akali pia ome manta se ʃoʃotik wanokse* the boat has two sails one green other *soltik* yellow (b) *in akali ka ipan atɬ* the boat is in water
13.		*non tɬaneʃti kapilkatok ipan tɬapetʃ* thing make.light hanging on table
15.		*in tapametɬ tɬatsakwa se kali* the fence/wall around the house
16.		*in tolontik ka tɬatsintsin ʃe tɬapetʃ* the ball is under a/one chair

	WordsEye Scene	Nahuatl description and gloss
17.		*in kwitɬapa tepetɬ se kwatli eltok* the back mountain a tree standing
		*in kwatli ka tɬapak in tepettɬ* the tree is top.of the hill/mountain
19.		*in kakatsi pia se mansana* the little.plate has one apple
21.		*in tɬakatɬ-sintɬi pia sa se tekak* the man-with.respect has only one shoe/sandal
22.		*jei amame kwakolonike ekan se akoʃa* three paper.PL make.a.hole.inside with a needle
24.		*in amatɬ tɬakentija se kutʃara* the paper covers one spoon
25.		*se weka.tɬatoa o-tɬalike ipan tpametɬ* one far.speak(phone) PST-put in wall
29.		*in mantana ka ipan tɬapetʃ* the cloth is on table

	WordsEye Scene	Nahuatl description and gloss
30.		*in kwawitɬ tɬapanawi tɬakoja se mansana* the stick pass.through in.middle one apple
31.		*in misto tɬatʃia tɬatsintɬa se tɬapetʃ* the cat watching under one table
33.		*in kwakwatsi tɬakentʃia se laso* the little.piece.of.wood biting one rope
34.		*in tɬakatɬ o-tɬeko ipan kali* the man PST-get.up on house
38.		*in osomatɬ mitotia inawak in tɬetɬ* the monkey dancing close.to the fire
39.		*in tɬakat-sintɬi tɬatʃitʃina* the man-with.respect smoking
40.		*in iskwintɬi jewatok ipan petɬatɬ* the dog sitting in mat
44.		*in tɬakwilo-tsin ka ipan tpametɬ* the painting-with.respect is in wall

	WordsEye Scene	Nahuatl description and gloss
47.		*se weij tʃitʃi ijewatok ipan itłapetʃ* one big dog sitting on his.bed
49.		*in kwatsi ka iʃpantsi in teopantsinko* the tree is in.front the church/building
		*in kwatsin ka ikwitłapa in teopantsinko* the tree is in.back the church
53.		*in tsopelik ka tsekotok tłatsintła in tłapetʃ* the candy is sticking under the table
54.		*in toʃtłi kejsa wan tʃolos* the rabbit going.out and running.away
56.		(a) *in pantłi ka inawak se kali* the flag is close.to the house (b) *in pantłi ka ipan se weij kwawitł itok* the flag is in one big stick standing
58.		*in eskaleda tʃitʃiltik ka ipan tepametł* the ladder red is on wall
59.		*in tłamakwilo ka tłakojo ipan tłapetʃ* the pencil is in.the.middle in table

	WordsEye Scene	Nahuatl description and gloss
60.		(a) *se weikali ka paltitok tɬapak istak wan* a house is painting top white and *tɬatsintɬla tɬiltik* in.bottom black (b) *in kali ka intek se tepametɬ* the house is inside a wall
62.		*in tɬatsakwal in tawilotɬ ka istak* the cap the bottle is white
64.		*in tɬakatɬ mot in tɬapetʃ* the man hiding the chair
		*in se wa-tsintɬi te moa itɬawekal* the one woman-with.respect is looking.for husband
65.		*ipan tepetɬ pia se kwakitɬ tɬapak* on hill have one tree top
68.		*in tɬakatɬ akok imawa* the man lift his.hands
69.		*in sewatɬ pia ipan in menakas se tepos* the woman has in the ear one metal *pilotok* hanging
71.		*in weiȷ tʃitʃi mosewia kaletek ikale* the big dog resting inside his.house

Appendix C

Arrernte Data: Spatial Language and Case

The data in this appendix consists of Arrernte sentences that we translated and glossed, with the help of Myfany Turpin, as described in Chapter 3, Section 3.8. The sentences we chose are interesting in terms of spatial language and/or case. The following sentences are originally from the *Eastern and Central Arrernte Picture Dictionary* [Broad, 2008].

(8) [**warle** 'house', page 24]

Akngwelye	*atherre*	*warle*	*arrwekele*	*anerle-anerreme*			
akngwelye	atherre	warle	arrwekele	ane	rlane	-rre	-me
kngwelye	atherre$_2$	warle	arrwekele	ane$_1$	-rl-[V^1][C^1][V^2]	-rre$_2$	-me$_1$
dog	two	house	front	sit	CONT	DU	NPP
n	num	n	n	v	v:ASP	v:NUM	v:TNS

'The two dogs are sitting at the front of the house.'

(9) [**ulye** 'shade', page 24]

Ulye	*aneme*		*uterne-ketye*		*anetyeke*	
ulye	ane	-me	uterne	-ketye	ane	-tyeke
ulye	ane$_1$	-me$_1$	uterne	-ketye	ane$_1$	-tyeke
shade	sit	NPP	sun	AVER	sit	PURP
n	v	v:TNS	n		v	v:

'The shade shelter gives protection from the sun.'

(10) [**alernnge, aherrke (NE), uterne** ‘sun’, page 27]

Artwe	*akaperte*	*artelheme*			*uternele*		*inngerre*
artwe	akaperte	arte	-lhe-	-me	uterne	-le	inngerre
artwe	akaperte	arte$_1$	-lhe-	-me$_1$	uterne	-le$_1$	inngerre
man	head	AVER	REFL	NPP	sun	ERG	face
n	n	v:	v>v	v:TNS	n	n:	n

ampeketye-nge		
ampe	-ketye	-nge
ampe$_2$	-ketye	-nge
burn	AVER	ABL
v	n:	n:

‘The man is putting his hat on so the sun doesn’t burn his face.’

(11) [**rlke** ‘wind’, page 27]

Arelhe	*rlkele-ureke*			*apetyeme*	
arelhe	rlke	-le	-ureke	apetye	-me
arelhe	rlke	-le$_3$	-ureke	apetye	-me$_1$
woman	wind	LOC	through	come	NPP
n	n	n:Any	n:Any	v	v:TNS

‘The woman is walking along through the wind.’

(12) [**apwape (C), urrepurrepe, arlewarrere** ‘whirlwind, willy-willy’, page 27]

Apwape	*akngerre*	*ularre*	*apetyeme*	
apwape	akngerre	ularre	apetye	-me
apwape	kngerre	ularre	apetye	-me$_1$
whirlwind	big	facing	come	NPP
n	adj	adp	v	v:TNS

‘A big willy-willy is coming this way.’

(13) [**arnerre** ‘rockhole’, page 29]

Arrekwelenye		*areyele*		*kwatye*	*antywetyarte*	
arrekwele	-nye	-areye	-le	kwatye	antywe	-tyarte
arrwekele	-nye	-areye	-le$_1$	kwatye	antywe$_1$	-tyerte
front	TMP.NOM	PL	ERG	water	drink	REM.P.HAB
n		n:Any	n:Any	n	v	v:TNS

arnerrenge-ntyele		
arnerre	-nge	-ntyele
arnerre	-nge	-ntyele$_1$
rockhole	ABL	ABL
n	n:Any	n:Any

‘In the old days, people used to drink water from rockholes.’

(14) [**atnye**me**, angerne**me **(C)** ‘dig a soakage’, page 29]

Arelhele		*ngentye*	*tyampitele*		*angerneme*		*kwatye*
arelhe	-le	ngentye	tyampite	-le	angerne	-me	kwatye
arelhe	-le$_1$	ngentye	tyampite	-le$_4$	angerne	-me$_1$	kwatye
woman	ERG	soakage	tin	INS	dig	NPP	water
n	n:Any	n	n	n:Any	v	v:TNS	n

antywetyeke	
antywe	-tyeke
antywe$_1$	-tyeke
drink	PURP
v	v:TNS

‘The woman is digging a soakage with a tin to get a drink of water.’

(15) [**ngentye** ‘soakage’, page 29]

Arelhele		*kwatye*	*ngentye*	*tyampitele*		*athankweme*	
arelhe	-le	kwatye	ngentye	tyampite	-le	athankwe	-me
arelhe	-le$_1$	kwatye	ngentye	tyampite	-le$_4$	athankwe	-me$_1$
woman	ERG	water	soakage	tin	INS	scoop	NPP
n	n:Any	n	n	n	n:Any	v	v:TNS

‘The woman is using a tin to scoop water from the soakage.’

(16) [**altyiwe**me ‘throw out, tip out, spill’, page 29]

Arelhele		*kwatye*	*altyiweme*		*ahelheke*	
arelhe	-le	kwatye	altyiwe	-me	ahelhe	-ke
arelhe	-le$_1$	kwatye	altyiwe	-me$_1$	ahelhe	-ke$_2$
woman	ERG	water	throw out	NPP	ground	DAT
n	n:Any	n	v	v:TNS	n	n:Any

‘The young woman is tipping water out onto the ground.’

(17) [**kwatyeke irrpe**me ‘swim, dive into water’, page 29]

Ampe	*atherre*	*kwatye*	
ampe	atherre	kwatye	-ke
ampe$_1$	atherre$_2$	kwatye	-ke$_2$
child (son or daughter of a woman)	two	water	DAT
n	num	n	n:Any

irrperle-anerreme			
irrpe	rlane	-rre	-me
irrpe	-rl-[V^1][C^1][V^2]	-rre$_2$	-me$_1$
go in	CONT	DU	NPP
v	v:ASP	v:NUM	v:TNS

‘The two kids are going for a swim.’

(18) [**kwatye** ‘water’, page 30]

Tapenge-ntyele			*kwatye*	*thelelheme*		
tape	-nge	-ntyele	kwatye	thele	lhe	-me
tape	-nge	-ntyele$_1$	kwatye	thele	-lhe-	-me$_1$
tap	ABL	ABL	water	pour	REFL	NPP
n	n:Any	n:Any	n	v	v>v	v:TNS

‘The water is pouring out of the tap.’

(19) [**alhewelhe**me**, alkngelhe**me **(N), urlkernelhe**me **(C)** ‘wash self’, page 30]

Marle	*nhenhe*	*iltye*	*kwatyele*		*alhewelheme*		
marle	nhenhe	iltye	kwatye	-le	alhewe	-lhe-	-me
marle	nhenhe	iltye	kwatye	-le$_4$	alhewe	-lhe-	-me$_1$
girl	this	hand	water	INS	wash	REFL	NPP
n	det	n	n	n:Any	v	v>v	v:TNS

‘This girl is washing her hands with water.’

(20) [**antywe**me ‘drink’, page 30]

Marle	*akwekele*		*kwatye*	*antyweme*	
marle	akweke	-le	kwatye	antywe	-me
marle	akweke	-le$_1$	kwatye	antywe$_1$	-me$_1$
girl	small	ERG	water	drink	NPP
n	adj	n:Any	n	v	v:TNS

‘The young girl is drinking water.’

(21) [**thelelhe**me**, altyiwelhe**me ‘pour out of, leak out’, page 30]

Tyampite	*altywere-akertenge*			*kwatye*	*altyiwelheme*		
tyampite	altywere	-akerte	-nge	kwatye	altyiwe	lhe	-me
tyampite	altywere	-kerte	-nge	kwatye	altyiwe	-lhe-	-me$_1$
tin	hole	PROP	ABL	water	throw out	REFL	NPP
n	n	n:Any	n:Any	n	v	v>v	v:TNS

‘Water is leaking out from the hole in the billycan.’

(22) [**mpwaltye** ‘type of frog’, page 31]

Mpwaltye	*ahelhe*	*kwenenge*		*artelhemele*			*aneme*	
mpwaltye	ahelhe	kwene	-nge	arte	-lhe-	-mele	ane	-me
mpwaltye	ahelhe	kwene	-nge	arte$_2$	-lhe-	-mele$_1$	ane$_1$	-me$_1$
type of frog	ground	underneath	ABL	bury	REFL	SS	sit	NPP
n	n	adp	n:Any	v	v>v	v:TNS	v	v:TNS

kwatyeke		*akarelhemele*		
kwatye	-ke	akare	-lhe-	-mele
kwatye	-ke$_2$	akare	-lhe-	-mele$_1$
water	DAT	wait	REFL	SS
n	n:Any	v	v>v	v:TNS

‘This kind of frog buries itself in the soil waiting for rain.’

(23) [**irrpennge (E, C), artepinye** ‘fish’, page 31]

Irrpennge	*kwatyenge-ntyele*			*akertne-irremele*		
irrpennge	kwatye	-nge	-ntyele	akertne	-irre-	-mele
irrpennge	kwatye	-nge	-ntyele$_1$	akertne	-irre-	-mele$_1$
fish	water	ABL	ABL	on top	INCH	SS
n	n	n:Any	n:Any	adj	v:Any	v:TNS

arrepe-iweme		*rlke*	*kwernetyenhele*		
arrepiwe	-me	rlke	kwerne	-tyenhe	-le
arrepiwe	-me$_1$	rlke	kwerne	-tyenhe	-mele$_1$
spit	NPP	wind	swallow	NPC	SS
v	v:TNS	n	v	v:TNS	v:TNS

‘Fish come up to the surface of the water to get air.’

(24) [**aherre** ‘kangaroo (*Macropus rufus*)’, page 32]

Kere	*aherre*	*akethele*		*anerleanemele*		
kere	aherre	akethe	-le	ane	rleane	-mele
kere	aherre	akethe$_2$	-le$_3$	ane$_1$	-rl-[V^1][C^1][V^2]	-mele$_1$
meat	kangaroo	in the open	LOC	sit	CONT	SS
n	n	adj	n:Any	v	v:ASP	v:TNS

arerle-aneme	*rleane*	
are	-rl-[V^1][C^1][V^2]	-me
are	CONT	-me$_1$
look	v:ASP	NPP
v		v:TNS

‘The kangaroo sits up and looks around the open plains.’

(25) [**arenge, apwerte-arenye** ‘euro, wallaroo (*Macropus robustus*)’, page 32]

Arenge	*apele*	*kere*	*apwertearenye*	
arenge	apele	kere	apwerte	-arenye
arenge	apele	kere	apwerte$_1$	-arenye
euro	FACT	meat	hill	ASSOC
n	prt	n	n	n:Any

‘The euro lives around the hills.’

(26) [**artnwere akwerrke** ‘pup’, page 32]

Artnwere	*akwerrke*	*areye*	*alhwengenge*		*arratewarreme*		
artnwere	akwerrke	-areye	alhwenge	-nge	arrate	-warre	-me
artnwere	akwerrke	-areye	alhwenge	-nge	arrate	-warre	-me$_1$
dingo	young	PL	hole	ABL	come out	PL	NPP
n	adj	n:Any	n	n:Any	v	n:Any	v:TNS

‘The dingo pups are coming out of the burrow.’

(27) [**kwarlpe** ‘hare wallaby (*Lagorchestes conspicillatus*)’, page 33]

Kere	*kwarlpe*	*atnarnpeme*		*apwertenge-ntyele*		
kere	kwarlpe	atnarnpe	-me	apwerte	-nge	-ntyele
kere	kwarlpe	atnarnpe	-me$_1$	apwerte$_1$	-nge	-ntyele$_1$
meat	hare wallaby	come down	NPP	hill	ABL	ABL
n	n	v	v:TNS	n	n:Any	n:Any

‘The hare wallabies make their way down from the hills.’

(28) [**antenhe** ‘possum (*Trichosurus vulpecula*)’, page 33]

Kere	*antenhe*	*artitye*	*arrirlpe-akerte*		*ingwele-ante*		
kere	antenhe	artitye	arrirlpe	-akerte	ingwe	-le	=ante
kere	antenhe	artitye	arrirlpe	-kerte	ingwe	-le$_3$	=ante
meat	possum	teeth	sharp	PROP	night	LOC	=ONLY
n	n	n	adj	n:Any	n	n:Any	prt

arrateme	
arrate	-me
arrate	-me$_1$
come out	NPP
v	v:TNS

‘Possums have sharp teeth and come out at night.’

(29) [**kamule** ‘camel’, page 34]

Kamule	*aneme*		*apmere*	*arrangkethe-arenye*	
kamule	ane	-me	apmere	arrangkethe	-arenye
kamule	ane$_1$	-me$_1$	apmere	arrangkethe	-arenye
camel	sit	NPP	country	bush	ASSOC
n	v	v:TNS	n	n	n:Any

‘Camels live out in the bush.’

(30) [**tangkeye** ‘donkey’, page 34]

Tangkeyele		*ularre*	*arerle-aneme*		
tangkeye	-le	ularre	are	rlane	-me
tangkeye	-le$_1$	ularre	are	-rl-[V^1][C^1][V^2]	-me$_1$
donkey	ERG	facing	look	CONT	NPP
n	n:Any	adp	v	v:ASP	v:TNS

‘The donkey is looking this way.’

(31) [**rapite** ‘rabbit’, page 35]

Rapite	*apele*	*alhwenge-arenye*	
rapite	apele	alhwenge	-arenye
rapite	apele	alhwenge	-arenye
rabbit	FACT	hole	ASSOC
n	prt	n	n:Any

‘Rabbits live in burrows.’

(32) [**karpele-karpele** ‘turkey’, page 35]

Karpele-karpele	*apele*	*kere*	*warlpele-kenhe*	
karpelekarpele	apele	kere	warlpele	-kenhe
karpelekarpele	apele	kere	warlpele	-kenhe
turkey	FACT	meat	white man	POSS
n	prt	n	n	n:Any

‘Turkeys are meat introduced by whitefellas.’

(33) [**arlpelhe (C), irlpelhe** ‘wing, feather’, page 36]

Nhenhe	*thipe-kenhe*		*arlpelhe*
nhenhe	thipe	-kenhe	arlpelhe
nhenhe	thipe	-kenhe	arlpelhe
this	bird	POSS	wing
det	n	n:Any	n

‘This is a bird’s wing.’

(34) [**antywe** ‘nest’, page 36]

Thipe	*antywele*		*anerle-aneme*		
thipe	antywe	-le	ane	rleane	-me
thipe	antywe$_2$	-le$_3$	ane$_1$	-rl-[V^1][C^1][V^2]	-me$_1$
bird	nest	LOC	sit	CONT	NPP
n	n	n:Any	v	v:ASP	v:TNS

‘The bird is sitting on the nest.’

(35) [**kwarte** ‘egg’, page 36]

Thipe-kenhe		*kwarte*	*antywe*	*kwenele*	
thipe	-kenhe	kwarte	antywe	kwene	-le
thipe	-kenhe	kwarte	antywe$_2$	kwene	-le$_3$
bird	POSS	egg	nest	inside	LOC
n	n:Any	n	n	adp	n:Any

‘The bird’s eggs are in the nest.’

(36) [**arrkarlpe** ‘bird’s crest’, page 36]

Arelhe	*arrkarlpe*	*akaperteke*		*arrernelheme*			*anthepeke*
arelhe	arrkarlpe	akaperte	-ke	arrerne	-lhe-	-me	anthepe
arelhe	arrkarlpe	akaperte	-ke$_2$	arrerne	-lhe-	-me$_1$	anthepe
woman	bird’s crest	head	DAT	put on	REFL	NPP	ceremony
n	n	n	n:Any	v	v>v	v:TNS	n

-ke
-ke$_2$
DAT
n:Any

‘Women put feathers from the bird’s crest on their heads for dancing.’

(37) [**ingkelthele** 'claw', page 36]

Irretye-kenhe		*ingkelthelele*		*thakwere*	*irrkweme*	
irretye	-kenhe	ingkelthele	-le	thakwere	irrkwe	-me
irretye	-kenhe	ingkelthele	-le$_4$	athakwere	irrkwe	-me$_1$
eagle	POSS	claw	INS	type of mouse	grab hold of	NPP
n	n:Any	n	n:Any	n	v	v:TNS

'The eagle's claw is grabbing hold of a mouse.'

(38) [**atnethwekethweke, urrilyare (SE, E)** 'button quail (*Turnix velox*)', page 37]

Atnethwekethweke	*name*	*kwenearenye*	
atnethwekethweke	name	kwene	-arenye
atnethwekethweke	name	kwene	-arenye
button quail	grass	inside	ASSOC
n	n	adp	n:Any

'The button quail lives among the grass.'

(39) [**tirre-tirre** 'rainbow bee-eater (*Merops ornatus*)', page 37]

Thipe	*tirre-tirre*	*lhere*	*arnkarrele*		*alhwengeke*	
thipe	tirre-tirre	lhere	arnkarre	-le	alhwenge	-ke
thipe	tirre-tirre	lhere	arnkarre	-le$_3$	alhwenge	-ke$_2$
bird	rainbow bee-eater	river	bank	LOC	hole	DAT
n	n	n	n	n:Any	n	n:Any

anenhe-anenhe	
ane	nheanenhe
ane$_1$	-nhe-[V^1][C^1][V^2]-nhe
sit	NMLZ.HAB.RDP
v	v>n

'The rainbow bee-eater lives in burrows on creek banks.'

(40) [**apelkere** 'crested pigeon (*Ocyphaps lophotes*)', page 39]

Apelkere	*apele*	*thipe*	*kwertearteke*		*anteme*	*akaperte*	*arrirlpe*
apelkere	apele	thipe	kwerte	=arteke	=anteme	akaperte	arrirlpe
apelkere	apele	thipe	kwerte	=arteke	=anteme$_1$	akaperte	arrirlpe
crested pigeon	FACT	bird	smoke	=SEMBL	and	head	sharp
n	prt	n	n	prt	conn	n	adj

'The crested pigeon is smoky coloured and has a crest.'

(41) [**kwepalepale** 'crested bellbird (*Oreoica gutturalis*)', page 40]

Kwepalepalele		*alkngarre-ilemele*		*arrpenhe-ketye*	
kwepalepale	-le	alkngarreile	-mele	arrpenhe	-ketye
kwepalepale	-le$_1$	alkngarreile	-mele$_1$	arrpenhe	-ketye
crested bellbird	ERG	warn	SS	another	AVER
n	n:Any	v	v:TNS	n	n:Any

'The crested bellbird warns you if something or someone is approaching.'

(42) [**teye-teye, rteye-rteye** ‘magpie-lark (*Grallina cyanoleuca*)’, page 40]

Teye-teye	*ahelhe*	*atertenge-ntyele*			*antywe*	*mpwareme*	
teye-teye	ahelhe	aterte	-nge	-ntyele	antywe	mpware	-me
teye-teye	ahelhe	aterte	-nge	-ntyele$_1$	antywe$_2$	mpware	-me$_1$
magpie-lark	ground	mud	ABL	ABL	nest	make	NPP
n	n	n	n:Any	n:Any	n	v	v:TNS

‘The magpie-lark makes its nest out of mud.’

(43) [**atyewaketye** ‘grey-crowned babbler (*Pomatostomus temporalis*)’, page 41]

Atyewaketye	*arne*	*atnartenge*	
atyewaketye	arne	atnarte	-nge
atyewaketye	arne	atnarte	-nge
grey-crowned babbler	plant, tree	under	ABL
n	n	adp	n:Any

anthepe-anthepe-irrentye				*akngerre*
anthepe	anthepe	irre	-ntye	akngerre
anthepe	-[...]	-irre-	-ntye$_1$	kngerre
ceremony	FREQ.RDP	INCH	NMLZ	big
n	n:Any	v:Any	v>n	adj

‘The grey-crowned babbler dances around under bushes.’

(44) [**pmwilyare** ‘bush thick-knee (*Burhinus grallarius*)’, page 44]

Pmwilyare	*ingwele-ante*			*arrateme*		*akethele*	
pmwilyare	ingwe	-le	=ante	arrate	-me	akethe	-le
pmwilyare	ingwe	-le$_3$	=ante	arrate	-me$_1$	akethe$_2$	-le$_3$
bush thick-knee	night	LOC	=ONLY	come out	NPP	in the open	LOC
n	n	n:Any	prt	v	v:TNS	adj	n:Any

unthentye-akngerre		
unthe	-ntye	akngerre
anthe	-ntye$_1$	kngerre
look for	NMLZ	big
v	v>n	adj

‘The bush thick-knee comes out at night and wanders on the plain.’

(45) [**awengkere, wengkere (C)** ‘Pacific black duck (*Anas superciliosa*)’, page 45]

Awengkerele		*ltare*	*awemele*		*kwatye*	*kweneke*		*atere*
awengkere	-le	ltare	awe	-mele	kwatye	kwene	-ke	atere
awengkere	-le$_1$	ltare	awe	-mele$_1$	kwatye	kwene	-ke$_2$	atere
pacific black duck	ERG	noise	hear	SS	water	inside	DAT	scared
n	n:Any	n	v	v:TNS	n	adp	n:Any	adj

irrpeme	
irrpe	-me
irrpe	-me$_1$
go in	NPP
v	v:TNS

‘When they hear a noise ducks dive under the water in fear.’

(46) [**atnarre-tharnke-tharnke** ‘black-tailed native hen (*Gallinula ventralis*)’, page 45]

Atnarre-tharnke-tharnke	*kwatye*	*iterele*		*anentye-akngerre*		
atnarre-tharnke-tharnke	kwatye	itere	-le	ane	-ntye	akngerre
atnarre-tharnke-tharnke	kwatye	itere	-le$_3$	ane$_1$	-ntye$_1$	kngerre
black-tailed native hen	water	beside	LOC	sit	NMLZ	big
n	n	adp	n:Any	v	v>n	adj

‘Black-tailed native hens live near the water.’

(47) [**ikwarre** ‘type of skink (*Ctenotus* spp.)’, page 47]

Ikwarre-kenhe		*alhwenge*	*lhere*	*iterele*	
ikwarre	-kenhe	alhwenge	lhere	itere	-le
ikwarre	-kenhe	alhwenge	lhere	itere	-le$_3$
type of skink	POSS	hole	river	beside	LOC
n	n:Any	n	n	adp	n:Any

anentye-akngerre		
ane	-ntye	akngerre
ane$_1$	-ntye$_1$	kngerre
sit	NMLZ	big
v	v>n	adj

‘This type of skink’s home is in a burrow beside the creek.’

(48) [**arntetherrke** ‘carpet snake (*Morelia bredli*)’, page 47]

Arntetherrke	*urternenge*		*arrateme*		*thakwere*
arntetherrke	urterne	-nge	arrate	-me	thakwere
arntetherrke	uterne	-nge	arrate	-me$_1$	athakwere
carpet snake	sun	ABL	come out	NPP	type of mouse
n	n	n:Any	v	v:TNS	n

arlkwenhe-arlkwenhe	
arlkwe	nhearlkwenhe
arlkwe	-nhe-$[V^1][C^1][V^2]$-nhe
eat	NMLZ.HAB.RDP
v	v>n

‘Carpet snakes come out in warm weather and they eat mice.’

(49) [**atwakeye** ‘wild orange (*Capparis mitchellii*)’, page 51]

Merne	*atwakeye*	*mpenge*	*arlenge*	*anthurrenge-ntyele*		
merne	atwakeye	mpenge	-arlenge	anthurre	-nge	-ntyele
merne	atwakeye	mpenge	-larlenge	anthurre	-nge	-ntyele$_1$
vegetable food	wild orange	ripe	COM	very	ABL	ABL
n	n	adj	n:Any	adv	n:Any	n:Any

ntyerneme	
ntyerne	-me
ntyerne	-me$_1$
smell	NPP
v	v:TNS

‘You can smell ripe wild oranges from a long way away.’

(50) [**inte, interlpe** ‘skewer’, page 52]

Nhenhe	*anaketye*	*intele*		*arrerneke*	
nhenhe	anaketye	inte	-le	arrerne	-ke
nhenhe	anaketye	inte$_1$	-le$_3$	arrerne	-ke$_1$
this	type of bush tomato	skewer	LOC	put	PC
det	n	n	n:Any	v	v:TNS

‘These bush tomato fruits have been skewered onto a stick.’

(51) [**atyankerne** ‘type of mistletoe (*Lysiana* spp. and *Amyema* spp.)’, page 53]

Merne	*atyankerne*	*artetyenge*		*lyapeme*	
merne	atyankerne	artetye	-nge	lyape	-me
merne	atyankerne	artetye	-nge	lyape	-me$_1$
vegetable food	type of mistletoe	mulga	ABL	grow	NPP
n	n	n	n:Any	v	v:TNS

ulpulpenge		*arrateme*	
ulpulpe	-nge	arrate	-me
ulpulpe	-nge	arrate	-me$_1$
spring	temporal	come out	NPP
n	n:Any	v	v:TNS

‘This type of mistletoe grows on mulgas in the spring.’

(52) [**arlatyeye** ‘pencil yam (*Vigna lanceolata*)’, page 54]

Merne	*arlatyeye*	*lhereke*		*atnyemele*	
merne	arlatyeye	lhere	-ke	atnye	-mele
merne	arlatyeye	lhere	-ke$_2$	atnye	-mele$_1$
vegetable food	pencil yam	creek	DAT	dig	SS
n	n	n	n:Any	v	v:TNS

inentye-akngerre	
ine	-ntyakngerre
ine	-ntyakngerre
get	always
v	v>n

‘You dig for pencil yams in the creek.’

(53) [**aperarnte** ‘sweet sap from gum trees’, page 55]

Ngkwarle	*aperarnte*	*renhe*	*apere*	*irrkngelhenge*
ngkwarle	aperarnte	renhe	apere	irrkngelhe
ngkwarle	aperarnte	renhe	apere	irrkngelhe
sweet things	sweet sap from gum trees	3.SG.ACC	river red gum	thin bark
n	n	pro	n	n

	ineme	
-nge	ine	-me
-nge	ine	-me$_1$
ABL	get	NPP
n:Any	v	v:TNS

‘Aperarnte is the sweet sap from under the thin bark of gum trees.’

(54) [**yerrampe (N)** ‘honeyant (*Camponotus* spp.)’, page 56]

Yerrampeke		*artetye-artetyeke*	*artetye*		*atnyeme*	
yerrampe	-ke	artetye	-[...]	-ke	atnye	-me
yerrampe	-ke$_2$	artetye	PL	-ke$_2$	atnye	-me$_1$
honeyant	DAT	mulga	n:Any	DAT	dig	NPP
n	n:Any	n		n:Any	v	v:TNS

‘You dig for honey ants where there are lots of mulgas growing.’

(55) [**pirtwerre, pirterre atnwengke** ‘plain pituri, sandhill pituri (*Nicotiana rosulata*)’, page 57]

Pirtwerre	*ahelhele*		*lyapentyeakngerre*	
pirtwerre	ahelhe	-le	lyape	-ntyakngerre
pirtwerre	ahelhe	-le$_3$	lyape	-ntyakngerre
pituri	dirt	LOC	grow	always
n	n	n:Any	v	v>n

‘Pitwerre grows on the plain country.’

(56) [**ingkwerlpe** ‘rock pituri (*Nicotiana gossei*)’, page 57]

Ingkwerlpe	*apwertele*		*lyapentyeakngerre*	
ingkwerlpe	apwerte	-le	lyape	-ntyakngerre
ingkwerlpe	apwerte$_1$	-le$_3$	lyape	-ntyakngerre
rock pituri	hill	LOC	grow	always
n	n	n:Any	v	v>n

‘Rock pituri grows on the hill country.’

(57) [**ntyere, antyere (C, NE)** ‘woolybutt grass (*Eragrostis eriopoda*)’, page 58]

Antyerenge		*irrtnye*	*ingkele*		*urlkernetyarte*	
antyere	-nge	irrtnye	ingke	-le	urlkerne	-tyarte
antyere	-nge	irrtnye	ingke	-le$_4$	urlkerne	-tyerte
woolybutt grass	ABL	skin	foot	INS	clean	REM.P.HAB
n	n:Any	n	n	n:Any	v	v:TNS

‘People used to clean the husks off wooly butt grass with their feet.’

(58) [**untyeyampe** ‘corkwood honey’, page 61]

Untyeyampe	*kwatyeke-arleke*		*arrernemele*		*antyweme*	
untyeyampe	kwatye	-karleke	arrerne	-mele	antywe	-me
untyeyampe	kwatye	-karleke	arrerne	-mele$_1$	antywe$_1$	-me$_1$
corkwood honey	water	ALL	put	SS	drink	NPP
n	n	n:Any	v	v:TNS	v	v:TNS

rlkerte-arle	
rlkerte	=rle
rlkerte	=rle$_1$
sick	=REL
adj	prt

‘You put corkwood honey in water and drink it when you are sick.’

(59) [**ahernenge** 'grub from river red gum (possibly *Trictena argentata*)', page 63]

Apere	*artekerrengentyele*			*tyape*
apere	artekerre	-nge	-ntyele	tyape
apere	artekerre	-nge	-ntyele$_1$	tyape
river red gum	root	ABL	ABL	edible grubs
n	n	n:Any	n:Any	n

ahernenge	*ineme*		*kwatye*	*iperre*
ahernenge	ine	-me	kwatye	-iperre
ahernenge	ine	-me$_1$	kwatye	-iperre
grub from river red gum	get	NPP	water	AFTER
n	v	v:TNS	n	n:Any

'You get grubs from the river red gum from the gum tree roots after rain.'

(60) [**ingwenenge** 'grub from branch of river red gum', page 63]

Ingwenenge	*apele*	*tyape*	*aperearenye*	
ingwenenge	apele	tyape	apere	-arenye
ingwenenge	apele	tyape	apere	-arenye
grub from branch of river red gum	FACT	edible grubs	river red gum	ASSOC
n	prt	n	n	n:Any

'Ingwenenge is the grub that lives in the branches of river red gums.'

(61) [**atyerre**me 'shoot', page 64]

Artwele		*makitele*		*atyerreme*		*kere*	*aherre*
artwe	-le	makite	-le	atyerre	-me	kere	aherre
artwe	-le$_1$	makite	-le$_4$	atyerre	-me$_1$	kere	aherre
man	ERG	rifle	INS	shoot	NPP	meat	kangaroo
n	n:Any	n	n:Any	v	v:TNS	n	n

tnerle-anerlenge		
tne	rlane	-larlenge
tne	-rl-[V^1][C^1][V^2]	-larlenge
stand	CONT	COM
v	v:ASP	n:Any

'The man is using a rifle to shoot the kangaroo standing there.'

(62) [**irrtyarte** 'spear', page 65]

Artwe	*nhenhe*	*irrtyarte*	*uthene*	*amirre*	*uthene-akerte*	
artwe	nhenhe	irrtyarte	-uthene	amirre	-uthene	-akerte
artwe	nhenhe	irrtyarte	-uthene	amirre	-uthene	-kerte
man	this	spear	also	woomera	also	PROP
n	det	n	conn	n	conn	n:Any

tnepe-tneme		
tne	petne	-me
tne	-pe[V^1][C^1][V^2]	-me$_1$
stand	FREQ.RDP	NPP
v	v:FREQ	v:TNS

'This man is standing there with his spears and woomera.'

(63) [**lerne**me ‘winnow seeds using a coolamon’, page 70]

Arelhe	*ampwe*	*nhenhele*		*urtnele*		*merne*	*ntange*
arelhe	ampwe	nhenhe	-le	urtne	-le	merne	ntange
arelhe	ampwe	nhenhe	-le$_1$	urtne	-le$_3$	merne	ntange
woman	old	this	ERG	coolamon	LOC	vegetable food	edible seeds
n	adj	det	n:Any	n	n:Any	n	n

lerneme	
lerne	-me
lerne	-me$_1$
winnow seeds using a coolamon	NPP
v	v:TNS

‘This old woman is winnowing the seeds in the coolamon.’

(64) [**aywerte** ‘spinifex (*Triodia* spp.)’, page 73]

Aywerte	*apwertele*		*tnentye-akngerreke*		
aywerte	apwerte	-le	tne	-ntyakngerre	-ke
aywerte	apwerte$_1$	-le$_3$	tne	-ntyakngerre	-ke$_2$
spinifex	hill	LOC	stand	always	DAT
n	n	n:Any	v	v>n	n:Any

ankere	*ineme*	
ankere	ine	-me
ankere	ine	-me$_1$
lump of spinifex resin	get	NPP
n	v	v:TNS

‘You get resin from spinifex, which grows on the hills.’

(65) [**urrknge-ile**me ‘make soft, soften’, page 73]

Aywertenge-ntyele			*ankere*	*inemele*	
aywerte	-nge	-ntyele	ankere	ine	-mele
aywerte	-nge	-ntyele$_1$	ankere	ine	-mele$_1$
spinifex	ABL	ABL	lump of spinifex resin	get	SS
n	n:Any	n:Any	n	v	v:TNS

artwele		*urrknge-ilemele*			*iteme*	
artwe	-le	urrknge	ile	-mele	ite	-me
artwe	-le$_1$	urrknge	-ile-	-mele$_1$	ite$_2$	-me$_1$
man	ERG	soft	CAUS	SS	cook	NPP
n	n:Any	adj	v:Any	v:TNS	v	v:TNS

‘The man got some spinifex resin and is cooking it to make it soft.’

(66) [**akngerne**me ‘carry, take back’, page 74]

Arelhe	*nhenhele*		*arlkethe*	*kwatyeakerte*		*akapertele*	
arelhe	nhenhe	-le	arlkethe	kwatye	akerte	akaperte	-le
arelhe	nhenhe	-le$_1$	arlkethe	kwatye	-kerte	akaperte	-le$_3$
woman	this	ERG	water dish	water	PROP	head	LOC
n	det	n:Any	n	n	n:Any	n	n:Any

akngerneme	
akngerne	-me
akngerne	-me$_1$
carry	NPP
v	v:TNS

'This woman is carrying a full water dish on her head.'

(67) [**artetye** 'mulga (*Acacia aneura*)', page 75]

Artetyenge-ntyele			*apele*	*arne*	*atningke*	*mpwarentye*
artetye	-nge	-ntyele	apele	arne	atningke	mpware
artetye	-nge	-ntyele$_1$	apele	arne	atningke	mpware
mulga	ABL	ABL	FACT	plant, tree	many	make
n	n:Any	n:Any	prt	n	quant	v

	akngerre
-ntye	akngerre
-ntye$_1$	kngerre
NMLZ	big
v>n	adj

'Lots of different things are made from mulga wood.'

(68) [**alye** 'boomerang', page 75]

Tyerrtyele		*urrenyenkenge-ntyele*			*alye*
tyerrtye	-le	urrenyenke	-nge	-ntyele	alye
tyerrtye	-le$_1$	urrenyenke	-nge	-ntyele$_1$	alye
person	ERG	gidgee tree	ABL	ABL	boomerang
n	n:Any	n	n:Any	n:Any	n

mpwaretyarte	
mpware	-tyarte
mpware	-tyerte
make	REM.P.HAB
v	v:TNS

'Aboriginal people used to make boomerangs from gidgee trees.'

(69) [**tyangaye** 'shanghai, slingshot', page 75]

Tyangaye	*arntarlkwe*	*akwekengentyele*		
tyangaye	arntarlkwe	akweke	-nge	-ntyele
tyangaye	arntarlkwe	akweke	-nge	-ntyele$_1$
shanghai	fork in tree	small	ABL	ABL
n	n	adj	n:Any	n:Any

mpwarentye-akngerre	
mpware	-ntyakngerre
mpware	-ntyakngerre
make	always
v	v>n

'Shanghais are made from a small, forked stick.'

(70) [**kwetere** ‘fighting stick, nulla-nulla’, page 75]

Kwetere	*arne*	*artetyenge-ntyele*			*mpwareme*	
kwetere	arne	artetye	-nge	-ntyele	mpware	-me
kwetere	arne	artetye	-nge	-ntyele$_1$	mpware	-me$_1$
fighting stick	plant, tree	mulga	ABL	ABL	make	NPP
n	n	n	n:Any	n:Any	v	v:TNS

arelhele		*ilpelhetyeke*		
arelhe	-le	ilpe	-lhe-	-tyeke
arelhe	-le$_1$	ilpe	-lhe-	-tyeke
woman	ERG	shield, protect	REFL	PURP
n	n:Any	v	v>v	v:TNS

‘Women use nulla-nullas made from mulga wood to protect themselves.’

(71) [**akwerne**me ‘put in, put something in something’, page 79]

Arelhele		*ingkwerlpe*	*ikwerenhe*	*yakwetheke*		*akwerneme*	
arelhe	-le	ingkwerlpe	ikwerenhe	yakwethe	-ke	akwerne	-me
arelhe	-le$_1$	ingkwerlpe	ikwerenhe	yakwethe	-ke$_2$	akwerne	-me$_1$
woman	ERG	tobacco	3.SG.POSS	bag	DAT	put in	NPP
n	n:Any	n	pro	n	n:Any	v	v:TNS

‘The woman is putting her tobacco into the bag.’

(72) [**arrerne**me ‘put’, page 79]

Arelhele		*tyampite*	*arrerneme*		*ureke-arleke*		
arelhe	-le	tyampite	arrerne	-me	ure	-ke	arleke
arelhe	-le$_1$	tyampite	arrerne	-me$_1$	ure	-ke$_2$	-rleke$_1$
woman	ERG	billycan	put	NPP	fire	DAT	AS.WELL
n	n:Any	n	v	v:TNS	n	n:Any	

urinpe-irretyeke		
urinpe	-irre-	-tyeke
urinpe	-irre-	-tyeke
hot	INCH	PURP
adj	v:Any	v:TNS

‘The woman is putting the billycan on the fire to heat it up.’

(73) [**apmerenge-apmere** ‘cubbyhouse’, page 80]

Ampe	*marle*	*atherre*
ampe	marle	atherre
ampe$_1$	marle	atherre$_2$
child (son or daughter of a woman)	girl	two
n	n	num

apmerengeapmereke		
apmere	ngeapmere	-ke
apmere	-nge[...]	-ke$_2$
home	RDP.play	DAT
n	n>n	n:Any

arrkerne-irrerleanerreme				
arrkerne	-irre-	-rl-[V^1][C^1][V^2]	-rre	-me
arrkerne	-irre-	-rl-[V^1][C^1][V^2]	-rre$_2$	-me$_1$
fun	INCH	CONT	DU	NPP
n	v:Any	v:ASP	v:NUM	v:TNS

'The two girls are playing cubbyhouse.'

(74) [**ingkele atwe**me 'kick', page 80]

Artwe	*nhenhele*		*tyaperapere*	*ingkele*		*atweme*	
artwe	nhenhe	-le	tyaperapere	ingke	-le	atwe	-me
artwe	nhenhe	-le$_1$	tyaperapere	ingke	-le$_4$	atwe	-me$_1$
man	this	ERG	football	foot	INS	hit	NPP
n	det	n:Any	n	n	n:Any	v	v:TNS

'This man is kicking the football.'

(75) [**amarrkele atnyene**me 'hold on the hip', page 81]

Mikwele		*ampe*	*urreye*	*akweke*	*amarrkele*
mikwe	-le	ampe	urreye	akweke	amarrke
mikwe	-le$_1$	ampe$_1$	urreye	akweke	amarrke
mother	ERG	child (son or daughter of a woman)	boy	small	hip
n	n:Any	n	n	adj	n

	atnyeneme	
-le	atnyene	-me
-le$_3$	atnyene	-me$_1$
LOC	hold	NPP
n:Any	v	v:TNS

'The mother is holding the little boy on her hip.'

(76) [**artepele aknge**me 'carry on back', page 81]

Mikwele		*artepele*		*akngeme*		*ampe*
mikwe	-le	artepe	-le	aknge	-me	ampe
mikwe	-le$_1$	artepe	-le$_3$	aknge	-me$_1$	ampe$_1$
mother	ERG	back	LOC	carry	NPP	child (son or daughter of a woman)
n	n:Any	n	n:Any	v	v:TNS	n

akweke-arle		*apurrke-irrerlenge*		
akweke	arle	apurrke	irre	-rlenge
akweke	=rle$_2$	apurrke	-irre-	-rlenge
small	REL	tired	INCH	DS
adj	prt	adj	v:Any	v:Any

'The mother is carrying her child on her back because he got tired.'

(77) [**untyele aknge**me ‘carry on shoulder’, page 81]

Arelhe	*nhenhele*		*ampe*	*urreye*	*akweke*
arelhe	nhenhe	-le	ampe	urreye	akweke
arelhe	nhenhe	-le$_1$	ampe$_1$	urreye	akweke
woman	this	ERG	child (son or daughter of a woman)	boy	small
n	det	n:Any	n	n	adj

untyele		*akngeme*	
untye	-le	aknge	-me
untye	-le$_3$	aknge	-me$_1$
shoulder	LOC	carry	NPP
n	n:Any	v	v:TNS

‘This woman is carrying the small boy on her shoulders.’

(78) [**atyerne**me ‘peep from behind something’, page 81]

Arelhe	*ampwele*		*warlenge*		*atyernemele*	
arelhe	ampwe	-le	warle	-nge	atyerne	-mele
arelhe	ampwe	-le$_1$	warle	-nge	atyerne	-mele$_1$
woman	old	ERG	house	ABL	peep from behind something	SS
n	adj	n:Any	n	n:Any	v	v:TNS

areme	
are	-me
are	-me$_1$
see	NPP
v	v:TNS

‘The old woman is peeping out from the side of the house.’

(79) [**arlpare-ane**me **(C, N), irlpareane**me ‘hang down’, page 83]

Marle	*nhenhe*	*arne*	*ngkwernenge*		*irlpare-aneme*	
marle	nhenhe	arne	ngkwerne	-nge	irlparane	-me
marle	nhenhe	arne	ngkwerne	-nge	irlparane	-me$_1$
girl	this	plant, tree	branch, trunk	ABL	hang down	NPP
n	det	n	n	n:Any	v	v:TNS

‘This girl is hanging down from the branch.’

(80) [**antye**me ‘climb’, page 83]

Urreye	*nhenhe*	*arneke*		*antyeme*	
urreye	nhenhe	arne	-ke	antye	-me
urreye	nhenhe	arne	-ke$_2$	antye	-me$_1$
boy	this	plant, tree	DAT	climb	NPP
n	det	n	n:Any	v	v:TNS

‘This boy is climbing up the tree.’

(81) [**atnye**me 'fall', page 83]

Urreye	*nhenhe*	*arne*	*akerntnengentyele*			*atnyeme*	
urreye	nhenhe	arne	akertne	-nge	-ntyele	atnye	-me
urreye	nhenhe	arne	akertne	-nge	-ntyele$_1$	atnye	-me$_1$
boy	this	plant, tree	on top	ABL	ABL	fall	NPP
n	det	n	adj	n:Any	n:Any	v	v:TNS

'This boy is falling down from up in the tree.'

(82) [**ularre apeyte**me 'come or face towards', page 85]

Awenhe-awenhe	*atyinhe*	*ularre*	*apetyeme*	
awenheawenhe	atyinhe	ularre	apetye	-me
awenhe-awenhe	atyenhe	ularre	apetye	-me$_1$
auntie (father's sister)	1.SG.POSS	facing	come	NPP
n	pro	adp	v	v:TNS

'My auntie is coming this way.'

(83) [**untye**me **alhe**me 'go back from or face away from', page 85]

Aperle-aperle	*atyinhe*	*untyemeatheke*		*alheme*	
aperleaperle	atyinhe	untyeme	-atheke	alhe	-me
aperleaperle	atyenhe	untyeme	-theke	lhe$_1$	-me$_1$
grandmother(FM)	1.SG.POSS	away	towards	go	NPP
n	pro	adv	adp	v	v:TNS

apmere-werneatheke		
apmere	-werne	-atheke
apmere	-werne	-theke
home	ALL	towards
n	n:Any	adp

'My grandmother is going back towards home.'

(84) [**anpere alhe**me 'go past', page 85]

Arelhe	*nhenhe*	*anpere*	*alheme*		*arenhemele*		
arelhe	nhenhe	anpere	alhe	-me	are	-nhe	-mele
arelhe	nhenhe	anpere	lhe$_1$	-me$_1$	are	-nhe$_2$	-mele$_1$
woman	this	past	go	NPP	see	DO.PAST	SS
n	det	adv	v	v:TNS	v	v:ASSOC.MOT	v:TNS

anerle-anerrerlenge			
ane	rl[V^1][C^1][V^2]	-rre	-rlenge
ane$_1$	-rl-[V^1][C^1][V^2]	-rre$_2$	-rlenge
sit	CONT	PL	DS
v	v:ASP	v:NUM	v:Any

'This woman is going past, looking at those two sitting down.'

(85) [**unte**me ‘run’, page 85]

Arelhe	*untyeme*	*unteme*		*iwenheketyenge*			*apeke*	*re*
arelhe	untyeme	unte	-me	iwenhe	-ketye	-nge	=apeke	re
arelhe	untyeme	unte$_2$	-me$_1$	iwenhe	-ketye	-nge	=apeke	re
woman	away	run	NPP	what	AVER	ABL	maybe	3.SG.NOM
n	adv	v	v:TNS	interrog	n:Any	n:Any	prt	pro

‘The woman must be running away from something.’

(86) [**artnerre ake**me**, artnerre**me ‘crawl’, page 85]

Ampe	*akweke*	*nhenhe*	*artnerre-akeme*	
ampe	akweke	nhenhe	artnerrake	-me
ampe$_1$	akweke	nhenhe	artnerrake	-me$_1$
child (son or daughter of a woman)	small	this	crawl	NPP
n	adj	det	v	v:TNS

mikwe-werne	
mikwe	-werne
mikwe	-werne
mother	ALL
n	n:Any

‘This baby is crawling toward her mother.’

(87) [**atnarnpe**me ‘get down, jump down’, page 85]

Meye	*mutekaye-ngentyele*			*atnarnpeme*	
meye	mutekaye	-nge	-ntyele	atnarnpe	-me
meye	mwetekaye	-nge	-ntyele$_1$	atnarnpe	-me$_1$
mother	car	ABL	ABL	come down	NPP
n	n	n:Any	n:Any	v	v:TNS

‘My mother is getting down out of the car.’

(88) [**irrpenhe**me ‘go in’, page 86]

Yaye	*apmere*	*kwene-werne*		*irrpenheme*	
yaye	apmere	kwene	-werne	irrpenhe	-me
yaye	apmere	kwene	-werne	irrpenhe	-me$_1$
elder sister	home	inside	ALL	go in	NPP
n	n	adp	n:Any	v	v:TNS

‘My sister is going into the house.’

(89) [**arratintye**me ‘come out’, page 86]

Yanhe	*atyenge-artweye*		*arriwenge*	
yanhe	atyenge	-artweye	arriwe	-nge
yanhe	atyenge	-artweye	arriwe	-nge
that(mid)	1.SG.DAT	custodian	doorway	ABL
det	pro		n	n:Any

arratintyeme		
arrat	-intye	-me
arrate	-intye	-me$_1$
come out	DO.COMING	NPP
v	V:ASSOC.MOT	V:TNS

‘That’s my relative coming out through the doorway.’

(90) [**pathekele** ‘bike’, page 86]

Urreye	*nhenhe*	*pathekele-nge*		*arrkene-irreme*		
urreye	nhenhe	pathekele	-nge	arrkene	-irre-	-me
urreye	nhenhe	pathekele	-nge	arrkerne	-irre-	-me$_1$
boy	this	bike	ABL	fun	INCH	NPP
n	det	n	n:Any	n	v:Any	v:TNS

‘This boy is having fun on a bike.’

(91) [**apure, apure-irre**me ‘shame, become embarrassed’, page 87]

Atyenge	*ipmenhe*	*apure-irreme*		
atyenge	ipmenhe	apure	-irre-	-me
atyenge	ipmenhe	apure	-irre-	-me$_1$
1.SG.DAT	grandmother(MM)	shame	INCH	NPP
pro	n	n	v:Any	v:TNS

‘My grandmother is feeling embarrassed.’

(92) [**arrangkwe** ‘no, nothing’, page 88]

Ayenge	*ngkweltyeke*		*arrangkwe*
ayenge	ngkweltye	-ke	arrangkwe
ayenge	ngkweltye	-ke$_2$	arrangkwe
1.SG.NOM	n	DAT	no
pro		n:Any	interj

‘I’ve got no money.’

(93) [**akweke** ‘small, little’, page 90]

Nhenhe	*unyerre*	*akweke-arle*	
nhenhe	unyerre	akweke	arle
nhenhe	unyerre	akweke	=rle$_2$
this	mountain devil	small	FOC
det	n	adj	prt

‘This is a small mountain lizard.’

(94) [**akngerre** ‘big’, page 90]

Nhenhe	*ankerrthe*	*akngerre-arle*	
nhenhe	ankerrthe	akngerre	arle
nhenhe	ankerrthe	kngerre	=rle$_2$
this	mountain lizard	big	FOC
det	n	adj	prt

‘This is a big mountain lizard.’

(95) [**arlpentye** ‘long’, page 90]

Merne	*arlatyeye*	*arrpenhe*	*areye*	*urteke*	*lyapeme*	*,*	*kenhe*
merne	arlatyeye	arrpenhe	-areye	urteke	lyape	-me	kenhe
merne	arlatyeye	arrpenhe	-areye	urteke	lyape	-me$_1$	kenhe
vegetable food	pencil yam	some	PL	short	grow	NPP	BUT
n	n	n	n:Any	adj	v	v:TNS	conn

arrpenhe	*areye*	*arlpentye-arle*	
arrpenhe	-areye	arlpentye	arle
arrpenhe	-areye	arlpentye	=rle$_2$
other	PL	long	FOC
pro-form	n:Any	adj	prt

‘Pencil yams grow to be either short or long.’

(96) [**arratyenye, arratye** ‘straight’, page 91]

Arne	*iterele*		*tnerle-aneme*			*iwerre*
arne	itere	-le	tne	-rl-[V^1][C^1][V^2]	-me	iwerre
arne	itere	-le$_3$	tne	-rl-[V^1][C^1][V^2]	-me$_1$	iwerre
plant, tree	beside	LOC	stand	CONT	NPP	road
n	adp	n:Any	v	v:ASP	v:TNS	n

arratyele	
arratye	-le
arratye	-le$_3$
straight	LOC
adj	n:Any

‘The tree is standing at the side of the straight road.’

(97) [**arlkenye-arlkenye, ultewarrerte** ‘stripy’, page 91]

Apmwe	*arlkenye-arlkenyeakerteke*			*arelhetyeke*		
apmwe	arlkenyearlkenye	-akerte	-ke	are	-lhe-	-tyeke
apmwe	arlkenye-arlkenye	-kerte	-ke$_2$	are	-lhe-	-tyeke
snake	stripy	PROP	DAT	see	REFL	PURP
n	adj	n:Any	n:Any	v	v>v	v:TNS

‘Watch out for that stripy snake.’

(98) [**nhakwe** ‘that, there (far distance)’, page 92]

Apwerte	*antherrtye*	*intwarre*	*nhakwe*	*inteme*	
apwerte	antherrtye	intwarre	nhakwe	inte	-me
apwerte$_1$	antherrtye	intwarre	nhakwe	inte$_2$	-me$_1$
hill	large rocky hill	other side	that, there (far distance)	lie	NPP
n	n	n	det	v	v:TNS

‘The hills are over there in the distance.’

(99) [**nhenhe** 'this, here', page 92]

Akngwelye	*nhenhe*	*anerle-aneme*		
akngwelye	nhenhe	ane	rlane	-me
kngwelye	nhenhe	ane$_1$	-rl-[V^1][C^1][V^2]	-me$_1$
dog	this	sit	CONT	NPP
n	det	v	v:ASP	v:TNS

'This dog is sitting.'

(100) [**yanhe, alertekwenhe** 'that, there (middle distance)', page 92]

Yanhe	*kere*	*aherre*	*atherre*	*ulyele*	
yanhe	kere	aherre	atherre	ulye	-le
yanhe	kere	aherre	atherre$_2$	ulye	-le$_3$
that(mid)	meat	kangaroo	two	shade	LOC
det	n	n	num	n	n:Any

tnerle-anerreme			
tne	rlane	-rre	-me
tne	-rl-[V^1][C^1][V^2]	-rre$_2$	-me$_1$
stand	CONT	DU	NPP
v	v:ASP	v:NUM	v:TNS

'Those two kangaroos are standing over there in the shade.'

(101) [**itwe, itere** 'beside, near', page 92]

Kere	*aherre*	*apwene*	*itwele*		*tnerleaneme*		
kere	aherre	apwene	itwe	-le	tne	rlane	-me
kere	aherre	apwene	itwe	-le$_3$	tne	-rl-[V^1][C^1][V^2]	-me$_1$
meat	kangaroo	cassia bush	beside	LOC	stand	CONT	NPP
n	n	n	adp	n:Any	v	v:ASP	v:TNS

'The kangaroo is standing near the cassia bushes.'

(102) [**akertne** 'on top', page 92]

Akngwelye	*yanhe*	*antyame*	*akertnele*	
akngwelye	yanhe	antyame	akertne	-le
kngwelye	yanhe	antyame	akertne	-le$_3$
dog	that(mid)	blanket	on top	LOC
n	det	n	adj	n:Any

interle-aneme		
inte	-rl-[V^1][C^1][V^2]	-me
inte$_2$	-rl-[V^1][C^1][V^2]	-me$_1$
lie	CONT	NPP
v	v:ASP	v:TNS

'The dog is lying on top of the blanket.'

(103) [**kwene** ‘underneath’, page 92]

Akngwelye	*yanhe*	*antyame*	*kwenele*	
akngwelye	yanhe	antyame	kwene	-le
kngwelye	yanhe	antyame	kwene	-le$_3$
dog	that(mid)	blanket	underneath	LOC
n	det	n	adp	n:Any

interle-aneme		
inte	-rl-[V^1][C^1][V^2]	-me
inte$_2$	-rl-[V^1][C^1][V^2]	-me$_1$
lie	CONT	NPP
v	V:ASP	V:TNS

‘The dog is lying underneath the blanket.’

(104) [**talkwe, tarlkwe** ‘across’, page 93]

Arne	*akngerre*	*yanhe*	*iwerrele*		*tarlkwe*	*inteme*	
arne	akngerre	yanhe	iwerre	-le	tarlkwe	inte	-me
arne	kngerre	yanhe	iwerre	-le$_3$	talkwe	inte$_2$	-me$_1$
plant, tree	big	that(mid)	road	LOC	across	lie	NPP
n	adj	det	n	n:Any	adp	v	V:TNS

‘That big log is lying across the road.’

(105) [**angathe, arrwekele** ‘this side of, in front of’, page 93]

Arelhe	*nhenhe*	*arne*	*arrekwele*	*tnerle-aneme*		
arelhe	nhenhe	arne	arrekwele	tne	-rl-[V^1][C^1][V^2]	-me
arelhe	nhenhe	arne	arrwekele	tne	-rl-[V^1][C^1][V^2]	-me$_1$
woman	this	plant, tree	front	stand	CONT	NPP
n	det	n	n	v	V:ASP	V:TNS

‘This woman is standing this side of the tree.’

(106) [**irntwarre, arrengakwe, ingkerne** ‘on the other side, behind’, page 93]

Arelhe	*yanhe*	*arnenge*		*arrengakwele*	
arelhe	yanhe	arne	-nge	arrengakwe	-le
arelhe	yanhe	arne	-nge	arrengakwe	-le$_3$
woman	that(mid)	plant, tree	ABL	behind	LOC
n	det	n	n:Any	adp	n:Any

tnerle-aneme		
tne	-rl-[V^1][C^1][V^2]	-me
tne	-rl-[V^1][C^1][V^2]	-me$_1$
stand	CONT	NPP
v	V:ASP	V:TNS

‘That woman is standing on the other side of the tree.’

(107) [**mpwepe, mpepe** 'in the middle, between', page 93]

Akngwelye	*artwe*	*nhenhe*	*atherre*	*mpwepele*		*aneme*	
akngwelye	artwe	nhenhe	atherre	mpwepe	-le	ane	-me
kngwelye	artwe	nhenhe	atherre$_2$	mpwepe	-le$_3$	ane$_1$	-me$_1$
dog	man	this	two	between	LOC	sit	NPP
n	n	det	num	adp	n:Any	v	v:TNS

'The dog is sitting in between the two men.'

(108) [**akertne** 'up high', page 93]

Arrkirlpangkwerle	*arrkernke*	*akertne*	*anthurrele*		*arlpare-aneme*	
arrkirlpangkwerle	arrkernke	akertne	anthurre	-le	arlpareane	-me
arrkirlpangkwerle	arrkernke	akertne	anthurre	-le$_3$	arlpare-ane	-me$_1$
bush coconut	gum tree	on top	very	LOC	hang down	NPP
n	n	adj	adv	n:Any	v	v:TNS

'The bush coconuts are up really high and hanging down from the gum tree.'

Appendix D

Primitive Spatial and Graphical Relations

This appendix lists the primitive spatial and graphical relations that were extracted from VigNet and used in the WordsEye semantics-to-scene system (see Chapter 5, Section 5.3.2). It also includes the spatial and graphical relations that were added to create the formal set of spatio-graphic primitives used by SpatialNet (see Chapter 6, Section 6.4.2). Spatial and graphical primitives that have been added as SGPs but are not currently supported by WordsEye are marked as *pending*.

The primitives have been organized into the following categories and sub-categories:

1. **Spatial relations:** • Position • Position (using affordances) • Orientation • Orientation+Position
2. **Size and shape:** • Dimensions • Shape • Shape+Position • Poses
3. **Part/whole relations:** • General • Regions • Collections
4. **Object properties:** • Graphical properties • Other physical properties • Scene properties • WordsEye object types

D.1 Spatial relations

Position

gfx.northeast-of.r

gfx.northwest-of.r

gfx.southwest-of.r

gfx.southeast-of.r

gfx.west-of.r

gfx.east-of.r

gfx.south-of.r

gfx.north-of.r

gfx.stage-right-of.r

gfx.stage-left-of.r

gfx.stage-back-of.r

gfx.stage-front-of.r

gfx.orthogonally-below.r

gfx.orthogonally-above.r

gfx.orthogonally-behind.r

gfx.orthogonally-in-front-of.r

gfx.orthogonally-right-of.r

gfx.orthogonally-left-of.r

gfx.above.r

gfx.level-with.r

gfx.below.r

gfx.near.r

gfx.next-to.r

gfx.behind.r

gfx.in-front-of.r

gfx.right-of.r

gfx.left-of.r

gfx.locally-at.r

gfx.laterally-centered-z.r

gfx.laterally-centered-y.r

gfx.laterally-centered-x.r

gfx.distance-from.r

gfx.embedded-in.r

(pending) gfx.position-between.r

(pending) gfx.in-surface.r

(pending) gfx.attached-to.r

Position (using affordances)

gfx.on-front-surface.r

gfx.on-top-surface.r

gfx.on-bottom-surface.r

gfx.in-3d-enclosure.r

(pending) gfx.on-oriented-surface.r

(pending) gfx.on-sloped-surface.r

(pending) gfx.on-peak.r

(pending) gfx.on-ridge.r

gfx.in-cup.r

gfx.stem-in-cup.r

gfx.under-canopy.r

gfx.in-entanglement.r

(pending) gfx.cap-on.r

(pending) gfx.cap-over.r

(pending) gfx.hang-from.r

(pending) gfx.connect-attachment-points.r

Orientation

gfx.upside-down.r	gfx.upright.r	gfx.orientation-with.r
gfx.supine.r	gfx.orientation-along.r	gfx.tilt-toward.r
gfx.face-up.r	gfx.orientation-from.r	*(pending)* gfx.axial-dir.r
gfx.face-down.r	gfx.orientation-toward.r	

Orientation+Position

(pending) gfx.cylinder-in-opening.r	*(pending)* gfx.lean-against.r
(pending) gfx.bridge-across.r	*(pending)* gfx.lean-on.r
(pending) gfx.protrude-from-surface.r	
(pending) gfx.protrude-from-axis.r	*(pending)* gfx.ring-around-cylinder.r
(pending) gfx.hole-through-sheet.r	*(pending)* gfx.planar-boundary-around.r

D.2 Size and shape

Dimensions

gfx.size.r	gfx.fit-depth.r	gfx.match-width.r
gfx.depth-of.r	gfx.fit-below.r	gfx.match-depth.r
gfx.width-of.r	gfx.fit-on-front-surface.r	*(pending)* gfx.fit-in.r
gfx.height-of.r	gfx.fit-on-top-surface.r	
gfx.3d-size-of.r	gfx.match-front-surface.r	*(pending)* gfx.fitted-on.r
gfx.fit-height.r	gfx.match-top-surface.r	*(pending)* gfx.cupped-size-of.r
gfx.fit-width.r	gfx.match-height.r	

Shape

(pending) gfx.shape-of.r	*(pending)* gfx.shape-of.loop.r
(pending) gfx.roundness-of.r	*(pending)* gfx.shape-of.folded.r
(pending) gfx.shape-bendiness-of.r	*(pending)* gfx.shape-of.folded-sheet.r
(pending) gfx.path.r	*(pending)* gfx.shape-of.folded-sheet-into-cylinder.r
(pending) gfx.open.r	
(pending) gfx.openness.r	*(pending)* gfx.shape-of.folded-cylinder.r

Shape+Position

(pending) gfx.sheet-draped-over.r
(pending) gfx.axial-strip-drape-over.r
(pending) gfx.cylinder-draped-over.r
(pending) gfx.sheet-conform-to-surface.r
(pending) gfx.cylinder-conform-to-surface.r
(pending) gfx.axial-strip-conform-to-surface.r
(pending) gfx.conforming-volume-around.r
(pending) gfx.coil-around.r
(pending) gfx.axial-strip-around.r

Poses

(pending) gfx.facial-expression.r
(pending) gfx.in-pose.r
(pending) gfx.move-joint-to.r
(pending) gfx.look-at.r
(pending) gfx.point-at.r
(pending) gfx.touch-with-body-part.r
(pending) gfx.hold-non-fixture.r
(pending) gfx.grasp-touch-fixture.r
(pending) gfx.hold.r
(pending) gfx.sit-on.r

D.3 Part/whole relations

[**]General

we.part-of.r
we.generic-part-of.r
we.named-part-of.r
gfx.spatial-tag.r

Regions

(pending) gfx.subregion.r
(pending) gfx.subregion-span.r
(pending) gfx.subvolume.r
(pending) gfx.subvolume-span.r
(pending) gfx.sublocation.r
(pending) gfx.sublocation-numeric.r
(pending) gfx.axial-cross-section.r
(pending) gfx.opening.r
(pending) gfx.interior-walls.r

[**]For relations in sub-categories marked with [**], WordsEye uses VigNet to find an existing object that already meets the specified constraints. These relations are not currently supported *graphically* by WordsEye.

Collections

we.member-of.r	*(pending)* gfx.aligned-group.r
we.group.r	*(pending)* gfx.volume-aggregation.r
(pending) gfx.oriented-group.r	*(pending)* gfx.planar-aggregation.r
(pending) we.sequence.r	*(pending)* we.part-collection.r

D.4 Object properties

Graphical properties

gfx.has-texture.r	gfx.reflectivity.r	gfx.visibility.r
gfx.rgb-value-of.r	gfx.brightness.r	
gfx.color-of.r	gfx.transparency.r	

****Other physical properties**

gfx.segmentation-axis.r	we.affordance-of.r	we.smoothness.r
gfx.length-axis.r	we.weight.r	we.hardness.r
gfx.preferred-surfaces.r	we.density.r	we.roughness.r
gfx.preferred-embeddedness.r	we.brittleness.r	we.stickiness.r
gfx.stretchable.r	we.resiliance.r	we.substance-of.r
gfx.3d-part-complexity.r	we.strength-as-resiliance.r	we.support-for.r

Scene properties

gfx.time-of-day.r	gfx.backdrop-receives-shadows.r

****WordsEye object types**

we.is-object.r	we.is-image.r	we.is-decal.r
we.is-cutout.r	we.is-panorama-image.r	we.is-abstract-object.r