An Architecture for Word Learning using Bidirectional Multimodal Structural Alignment

Keith Bonawitz
bonawitz@mit.edu

Anthony Kim
tkim42@mit.edu

Seth Tardiff
stardiff@mit.edu

MIT Artificial Intelligence Lab
545 Technology Square
Cambridge, MA 02139

Abstract

Learning of new words is assisted by contextual information. This context can come in several forms, including observations in non-linguistic semantic domains, as well as the linguistic context in which the new word was presented. We outline a general architecture for word learning, in which structural alignment coordinates this contextual information in order to restrict the possible interpretations of unknown words. We identify spatial relations as an applicable semantic domain, and describe a system-in-progress for implementing the general architecture using video sequences as our non-linguistic input. For example, when the complete system is presented with "The bird dove to the rock," with a video sequence of a bird flying from a tree to a rock, and with the meanings for all the words except the preposition "to," the system will register the unknown "to" with the corresponding aspect of the bird's trajectory.

1 Introduction

Multimodal word learning can be viewed as a problem in which inputs are presented concurrently in both the linguistic domain and at least one non-linguistic semantic domain. It is the responsibility of the word learner to (1) infer the correspondence of each word to some fragment of the semantic domain, and (2) refine the model of the word based on this correspondence (by generalizing the word semantics, for example).

In this paper[1], we propose a system aimed at the first half of the problem: inferring a correspondence between words and non-linguistic semantic domain fragments. In particular, we are interested in using the context of the word's introduction to limit possible interpretations. Assuming linguistic inputs are syntactic multiword utterances (e.g. phrases and sentences), this context includes the semantic and syntactic relationship of the new word to other words in the linguistic input. In a multimodal learning environment, the context also includes input observed in the non-linguistic domain. Our system is designed to coordinate all of these contextual clues in order to restrict the set of possible interpretations of the new word. By leveraging previously learned words to enable the learning of new words, we create a bootstrapping system for word learning.

In Section 2, we outline a general architecture based on symbolic structural alignment (Gentner and Markman, 1997) for solving the stated problem. In this outline, we identify necessary subsystems and requirements the system must satisfy. In Section 3, we identify the visual domain of spatial relations as a potential semantic domain, and in Sections 4–6 we describe a system-in-progress which instantiates the general architecture for this semantic domain.

2 General Architecture

We propose an architecture to answer the following question: assuming that a new word is embedded in a phrase with other previously acquired words, how can we exploit this linguistic context to focus on the fragments of non-linguistic input most likely to correspond with the new word?

The semantic principle of compositionality states that the meaning of any expression (such as a phrase) is a function of the meaning of its sub-expressions, where the particular function is determined by the method of composition. For example, the expression "The quick fox jumped over the log" can be considered a composition of the sub-expressions "The quick

[1]This research is supported, in part, by the National Science Foundation, Award Number IIS-0218861.

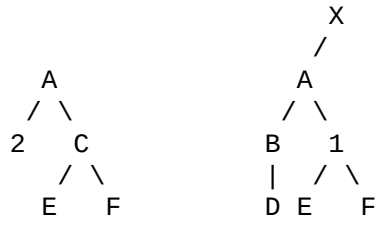

Figure 1: Structural alignment between these two structures infers the correspondences C↔1 and 2↔B(D). Structural alignment between semantic representations will bring unknown words into correspondence with their probable semantics.

fox" and "over the log", where syntactic composition with "jumped" is the method of composition. In other words, the semantics of this sentence can be expressed: $jumped$(SEMANTICS("The quick fox"), SEMANTICS("over the log")). Recursive application of this principle reveals that the semantic value of an expression is a structured representation.

Intuitively, then, we can approach our bootstrapping problem by structural alignment (Gentner and Markman, 1997). Structural alignment is a process in which corresponding elements in two structured representations are identified by matching. Correspondence between non-matching elements is then implied by the structural constraints of the representations.

For example, in Figure 1, structural alignment first matches A, E, and F between the two representations. Then, based on structural constraints, 1 is inferred to correspond with C, and 2 with B(D). In our architecture, structural alignment of the semantics of known words (and linguistic constituents formed thereof) with semantic structures observed in the non-linguistic domain will cause an alignment of unknown words with probable corresponding semantic fragments, thereby achieving our word learning goal of exploiting linguistic context to focus on fragments of the semantic input.

The remainder of this section describes the representations and methods required by a system seeking to implement this general architecture.

2.1 Semantic Representation

In order to perform structural alignment, the representation for the semantic domain must have several key properties:

- The domain must be a structural representation and it must be symbolic, in order to allow alignment of symbols.
- For inferences made from structural alignment to be valid, the representation must obey the principle of compositionality.
- The representation should contain orthogonal elements (i.e. the same piece of semantics is not encoded into multiple symbols) so that there are canonical ways of expressing particular meanings.
- Finally, the semantic representation must be lexicalized, implying that the semantics of any linguistic phrase can be cleanly divided amongst the phrase's constituent words. Each word should get a single connected semantic structure that does not share semantic symbols with any other word.

2.2 Semantic Processing

It is likely that the actual non-linguistic input modality will not be an appropriate structured symbolic representation. For example, the visual, aural, and kinesthetic modalities are non-symbolic. In any system dealing with such an input modality, it will be necessary to have modules that extract structured symbolic representations from the unstructured input.

2.3 Linguistic Processing

One challenge in performing structural alignment against language input is that the structured semantic representation of the linguistic input is implicit rather than explicit. Therefore, we need methods for parsing and an appropriate grammar. The grammar and parsing algorithms we choose must support several non-standard features.

First, we expect to encounter word meanings which are unknown, so our selected techniques must support gaps in the parse. We also require a reversible grammar, so that, when presented with the meaning of an entire expression and the meaning of some of its subexpressions, we can infer the meaning of the remaining subexpressions.[2]

Although it may not be required, parsing techniques that use partial structural alignment are preferred. Words and phrases have many possible interpretations and this problem is exacerbated by unknown words in the linguistic input. Since targets for the parse are available in the semantic input domain, use of these targets to guide the search through the space of possible linguistic interpretations is advantageous. Increasing structural alignment between the parsed semantics and the input semantics could be such a guiding heuristic. As a side effect of using structural alignment as a parsing heuristic, we should expect the parser to manipulate partial semantic and syntactic structures throughout the parsing process, as opposed to generating semantics from a completed syntactic parse tree after parsing is completed.

[2]Mathematically, this is equivalent to saying that there will be cases where we know a, f, and x in the equality $a = f(x, y)$, and we want to be able to infer the value of y. In order to do so, we must be able to compute the functional inverse of f with respect to y. That is, we want the function f_y^{-1} such that $y = f_y^{-1}(x, a)$.

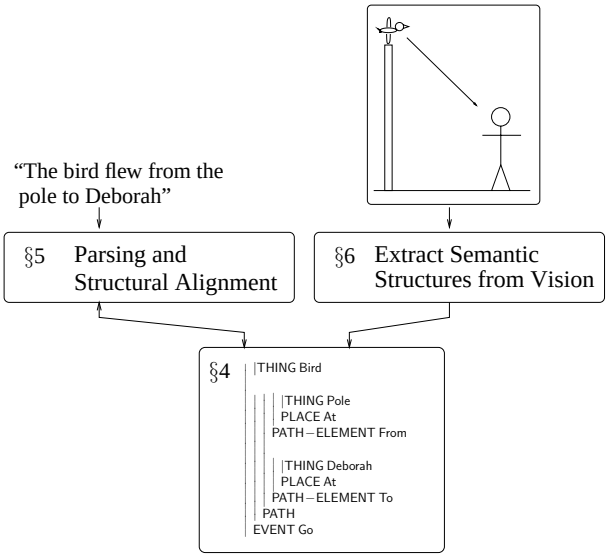

Figure 2: General system structure of our proposed implementation. Our learning-enabling structure is based on a semantic representation (Section 4) which is obtained by translating video inputs (Section 6). We then use a bidirectional search process to parse the linguistic input and to structurally align linguistic semantics with the non-linguistic semantics. (Section 5).

2.4 Structural Alignment

Gentner and Markman (1997) describe the requisite components of a structural alignment system as (1) methods for matching structural atoms, (2) methods for identifying sets of compatible atom matches (for example, ruling out cases in which two atoms in one structure map to the same atom in another structure), and (3) methods using atom matches to guide the matching of large portions of structure.

3 Implementation with Visual Domain of Spatial Relations

In order to validate our general architecture, we outline a system-in-progress which instantiates the architecture for the particular semantic domain of spatial relations. The domain of spatial relations captures the relative positioning, orientation, and movement of objects in space. Examples of sentences capturing spatial semantics include "The boy threw the ball into the box on the table", "The path went from the tree to the lakeside", and "The sign points to the door".

The following sections describe the methods and representations we have chosen to satisfy the requirements outlined in Section 2. Figure 2 shows how the system is designed and how the rest of this paper is organized.

Figure 3: Lexical-Conceptual Semantics is our semantic domain representation. This structure is the LCS model of "The bird flew from the pole to Deborah."

4 Lexical-Conceptual Semantics

We use Lexical-Conceptual Semantics (LCS) (Jackendoff, 1983) as our semantic representation. LCS is a cognitive representation that focuses on trajectories and spatial relations. Unlike other representations such as Logical Form (LF) and Conceptual Dependency (CD), LCS delineates notions of PATHs and PLACEs. LCS is more formally outlined in (Jackendoff, 1983) and is compared to other semantic representations in (Bender, 2001).

The following productions yield a simplified portion of the LCS space. For a complete description, refer to (Jackendoff, 1983).

- [THING]
- [PLACE] ⟵ PLACE-FUNC([THING])
  where PLACE-FUNC $\in$ {AT, ABOVE, BELOW, ON, IN, etc.}
- [PATH] ⟵ PATH(PATH-ELEMENT, PATH-ELEMENT, $\cdots$)
- [PATH-ELEMENT] ⟵ PATH-FUNC([PLACE])
  where PATH-FUNC $\in$ {TO, FROM, TOWARD, AWAY-FROM, VIA, etc.}
- [EVENT] ⟵ GO([THING],[PATH])
- [STATE] ⟵ BE([THING],[PLACE])
- [CAUSE] ⟵ CAUSE([THING],[EVENT])
- [CAUSE] ⟵ CAUSE([EVENT],[EVENT])

Using LCS, the trajectory expressed in the sentence "The bird flew from the pole to Deborah," is represented as in Figure 3

Lexical-Conceptual Semantics focuses on spatial relations in the physical world. However, it is easily extensible to other domains, such as the temporal and possessive domains (Jackendoff, 1983; Dorr, 1992). Research

focusing on using LCS in the abstract domain of social politics is also ongoing in our lab. Furthermore, it seems that much of language is spatial in nature. For example, there is significant psychological evidence that humans use spatial relations to talk about abstract domains such as time (Boroditsky, 2000). As a consequence, we believe that techniques for learning Lexical-Conceptual Semantics for words, developed here using the concrete spatial relations domain, will be extendible to many other domains.

5 Language Parsing and Structural Alignment

This section will describe the methods used to simultaneously parse linguistic input strings and align the resultant semantic structures with those from vision. The primary architecture of the system is a constraint propagation network using grammatical rules as constraints. A custom constraint network topology is generated for each linguistic input string using a bidirectional search algorithm.

5.1 Parsing/Alignment as Constraint Propagation

The parsing and structural alignment system may be viewed as a single large constraint, as in Figure 4. This constraint has two inputs: on one side, it takes a set of semantic representations originating from the vision processor. On the other side, it takes a linguistic input string, together with possible meanings for each word in the string, as determined by a lexicon (treating unknown words as having any possible meaning). As output, the constraint eliminates, in each input set, all meanings which do not lead to a successful structurally aligned parse.

In order to achieve such a complicated constraint, it is useful to decompose the constraint into a network of simpler constraints, each working over a local domain of only a few constituents rather than over the domain of an entire sentence, as in Figure 5. We can then base these subconstraints on grammatical rules over a fixed number of constituents, and trust the composed network to handle the complete sentence.

5.2 Grammatical Framework for Constraints

The grammar framework chosen for our system is Combinatorial Categorial Grammar (CCG) (Steedman, 2000). CCG has many advantages in a system like ours. First, there are only a handful of rules for combining constituents, and these rules are explicit and well defined. These qualities facilitate the implementation of constraints. In addition, CCG is adept at parsing around missing information, because it was designed to handle

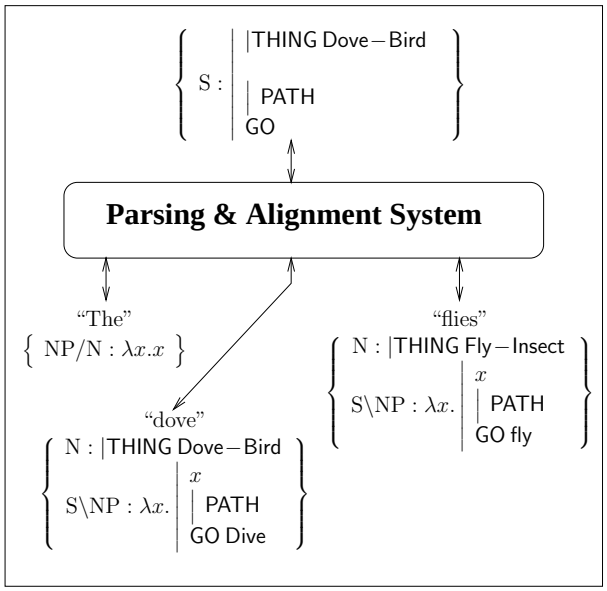

Figure 4: The parsing and structural alignment system functions as a constraint on linguistic and visual interpretations, requiring that expressions follow grammatical rules and that they align with the semantic domain input. This example shows the system presented with the sentence "The dove flies," and with a corresponding conceptual structure (from vision). In this situation, all the words were known, so the system will simply eliminate the interpretations of "dove" as a GO and flies as THINGs. If the word "dove" had not been known, the system would still select the verb form of flies (by alignment), which brings "dove" into alignment with the appropriate fragment of semantic structure (the THING frame).

linguistic phenomena such as parasitic-gapping[3]. The ability to gracefully handle incomplete phrases is crucial in our system, because it enables us to parse around unknown words.

In CCG, syntactic categories can either be atomic elements, or functors of those elements. The atomic elements are usually {S, N, NP} corresponding to Sentence, Noun, and Noun Phrase. Syntactic category functors are expressed in argument-rightmost curried notation, using / or \ to indicate whether the argument is expected to the left or right, respectively. Thus NP/N indicates a NP requiring a N to the right (and is therefore the syntactic category of a determiner), while (S\NP)/NP indicates an

[3]An example of a sentence with parasitic gapping is "John hates and Mary loves the movie," where both verbs share the same object. CCG handles this by treating "John hates" and "Mary loves" as constituents, which can then be conjoined by "and" into a single "John hates and Mary loves" constituent (traditional grammars are unable to recognize "John hates" as a constituent.)

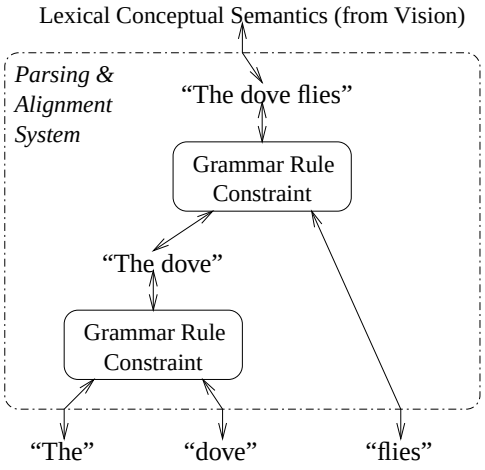

Figure 5: The parsing and alignment constraint in Figure 4 is actually implemented as a network of simpler constraints, as shown here. Each constraint implements a grammatical rule, as shown in Figure 6. The topology of the constraint network is dependent on the particular linguistic string, and is constructed by bidirectional search, as described in Section 5.3.

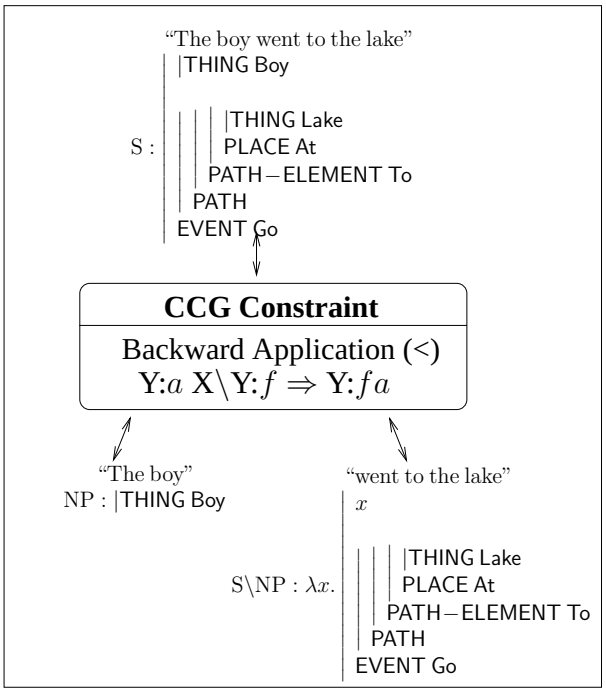

Figure 6: This figure shows one of the CCG Rules, Forward Functional Application, being treated as a constraint in a constraint propagation network. Any one of the three inputs can be left unspecified, and the constraint can completely determine the value based on the other two inputs.

S requiring one NP to the left and one to the right (this is the category of a monotransitive verb). For semantics, the notation is extended to X:f, indicating semantic category X with lambda calculus semantics f. These elements combine using a few simple productions, such as the following functional application rules[4]:

X/Y:f Y:a $\Rightarrow$ X:fa ($>$ *forward application*)
Y:a X$\backslash$Y:f $\Rightarrow$ X:fa ($<$ *backward application*)

5.3 Constructing the Constraint Network

The constraints we use are parse rules; therefore our constraint network topology embodies a parse tree for any sentence it can handle. Since our inputs do not include the parse tree, we must consider how to generate an appropriate constraint network topology.

One option is to use the same network topology to handle all sentences of the same length. Such a network would have to contain every possible parse tree, and thus would essentially result in an exhaustive search of the parse space. A better solution would be to avoid the exhaustive search by constructing a custom constraint topology for each sentence, using standard heuristic parse techniques. The drawback to this approach is that we are not interested in finding just any potential parse of a phrase/sentence, nor even the most statistically probable parse. Since our intent is to perform structural alignment with input from the non-linguistic domain, our goal in parsing is to find the semantic parse structure which aligns best with the semantic structure input from the non-linguistic domain. It follows that we should use the non-linguistic input to guide our search.

Our system applies bidirectional search to the parse/alignment problem. In contrast to traditional search techniques, bidirectional search treats both the "source" and "goal" symmetrically; the search-space is traversed both forward from the source and backward from the goal. The search processes operating in each direction interact with each other whenever their paths reach the same state in the search-space. This interaction provides hints for quickly completing the remainder of the search. For example, if the forward and backward paths reach the same search-state, then the forward searcher quickly reaches the goal by tracing the backward-path.

The specific style of bidirectional search we are investigating is based on Streams and Counterstreams (Ullman, 1996), in which forward and backward search paths interact with each other by means of primed pathways. For each transition, two priming values are maintained: a forward priming and backward priming. Primings are used when a decision must be made between several possible transitions that could extend a search path; those transitions that have a higher priming (using the for-

[4]For a full description and analysis of CCG, see (Steedman, 2000)

ward priming for forward searches, backward priming for backward searchers) are preferred for expansion. Transition primings in a particular direction (either forward or backward) are increased whenever a search path traverses the transition in the *opposite* direction. The net influence of the primings is that transitions previously traversed in one direction are more likely to be explored in the opposite direction, if the opportunity arises. By extension, primings provide clues for finding a path from any state to the target state.

The Streams and Counterstreams approach to bidirectional search facilitates incorporation of other types of context. For example, some situational context can be captured by allowing primings from previous parses of recent sentences to influence the current parse. Also, statistical cues such as Lexical Attraction (Yuret, 1999) can be integrated into the system by using heuristics to bias primings.

5.4 Structural Alignment

The three components of structural alignment specified in Section 2.4 (atomic alignment, identification of compatible match sets, and structurally implied matches) are woven into the bidirectional search construction of the constraint network topology. When a constraint network fragment is constructed which bridges between a small portion of the linguistic input and the non-linguistic semantics, this "atomic alignment" primes the bidirectional search to be more likely to repeat this match while constructing larger constraint network fragments; hence atomic alignment leads to larger structural alignment. The constraints in the constraint network ensure that all active atomic alignments are compatible. Finally, when the constraint network bridges large portions of the linguistic and non-linguistic inputs, the non-linguistic semantic structure gets partitioned across the words in the linguistic input by the grammatical constraints. This completes the structural alignment by bringing unknown words into correspondence with their probable semantics.

5.5 Handling Uncertainty

Throughout this discussion, we have considered words as being either completely learned or completely unlearned. Clearly, though, there is much middle ground, including words whose meanings are still ambiguous among several options, as well as words for which some meanings have been well acquired, while other valid meanings have yet to be learned. How can our system handle this degree-of-acquisition continuum?

Let us consider what we can expect from the meaning-refinement module. First, it should be able to report a set of witnessed possible meanings for each word, together with a correctness strength for each interpretation. This would be based on how regularly that interpretation has been witnessed. Furthermore, the module should be able to report the likelihood that the word still has unwitnessed interpretations; for initial occurrences of a word, this likelihood would be quite high, but with more exposure to the word, this likelihood would fall off.[5]

Returning to our system, we can now treat each word as having a set of known meanings together with a wildcard unknown meaning. When using bidirectional search to construct the constraint network topology, we bias the primings of transitions which reduce a word's potential meaning set, using the likelihood estimates given by the meaning-refinement module.

6 Lexical-Conceptual Structures from Video

The proposed system includes a vision component that is responsible for converting pixel data from a video input into the semantic structure described in Section 4. This vision system is an implementation of the ideas presented by John Bender (2001). Following Bender's prescriptions, the vision system does not perform object recognition. Instead, the goal of the system is to analyze the different paths and places that are present in a scene and, by relating these paths and places to one another, to construct an LCS representation of the actions.

6.1 Data Flow

The vision system consists of two parts. First, video frames are analyzed in sequence and the objects present in each scene are tracked using traditional vision algorithms and techniques[6]. For each object, information about the object's size, shape, and position over the life of the scene is stored in a data structure that we call a *Blob*. This name was chosen to highlight the fact that the vision system makes no attempt at object recognition or fine-grained analysis and is instead concerned only with paths along which the objects (blobs) move.

Second, the data regarding each object's progression through the scene is interpreted by an implementation of Bender's algorithm *DESCRIBE v.2* to produce the semantic representation that is used by the other components of the system.

6.2 Pixels to Blobs

The low-level portion of the vision system is fed sequences of pixel matrices by an external system that captures video data. In the current implementation, this pixel

[5] These likelihood estimations could be generated, among other ways, by a meaning-refinement module incorporating a Bayesian model.

[6] For details concerning image labeling and object extraction algorithms see (Horn, 1986)

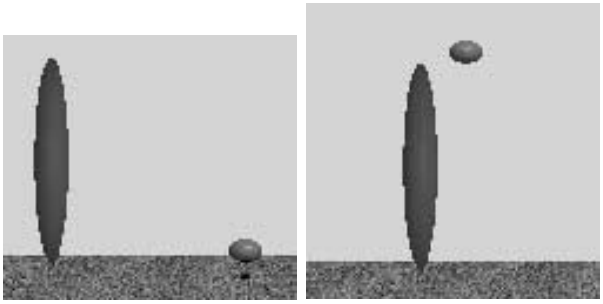

Figure 7: Start and end states of an example scene produced by the simulator. The left image represents the start state and the right image represents the final state.

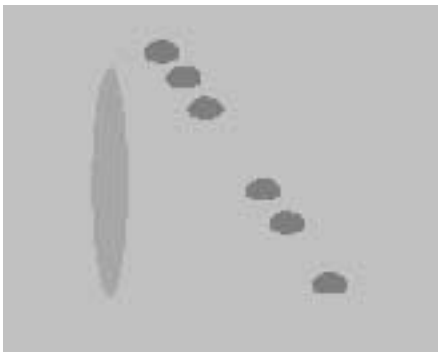

Figure 8: Trace of objects during the example scene. The moving object's position changes are tracked and the trace of its path is generated.

data is sent from a simulator in which the actions of simple objects take place. The pixel matrices include integer definitions of each pixel's value, supplying all color information.

When the analysis of a particular scene begins, the vision system captures a snapshot of the background that it uses as a reference for all subsequent frames related to the same scene. As new video frames are input, the stored background is subtracted and the new video frames are converted to binary images. A noise removal algorithm is applied to the binary images to remove any residual elements of the original background.

Once converted to a binary representation, each video frame is labeled using an object labeling algorithm and each distinct object is identified. Each object present within a frame is overlaid with a shape that will be used in the Blob representation passed along to the next component of the system. Each of the overlaid shapes is (possibly) matched to a shape observed in a previous frame. This matching procedure attempts to identify objects persisting between frames based on proximity in size, shape, color, and position using a 4-dimensional nearest-neighbors approach. If a shape matches with a previously known entry, the Blob structure corresponding to that particular object is assigned a new shape for its progression. If no match is found, a new Blob structure is created for the newly-observed object.

Once the analysis of all frames of a scene is complete, the list of Blobs is fed to the next portion of the vision system for further interpretation.

Figures 7 and 8 show an example of this portion of the vision system in use. Figure 7 shows the raw images representing the start and end states of the scene. Figure 8 shows a visualization of the object data created by the low-level portion of the system. The trace represents the path along which the object moved during the scene.

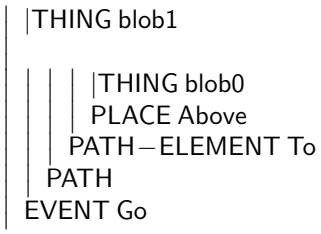

Figure 9: LCS frame produced by the vision system based on the example scene presented in Figure 7. Note that no object recognition is in use, so the objects are given temporary names (*blob0* and *blob1*).

6.3 Blobs to LCS

The generation of semantic structures from vision data concludes with an analysis of the Blobs generated by the low-level vision system. This analysis is performed by implementing an algorithm described, but never implemented in a system, by Bender (2001).

The algorithm first examines the list of objects present in the scene and computes the simple *exists?* and *moving?* predicates. If an object is found and moving, an LCS GO frame is instantiated and the object is compared to all others present so the appropriate path and place functions can be calculated. The calculation of path and place functions is based on a set of routines suggested by Bender. These routines compute the direction, intersection, and place-descriptions (above, on, left-of, etc.) for each pair of objects. Finally, the path and place functions described in Section 4 are found by examining the output of the visual routines and are added to the LCS frame.

Figure 9 shows the LCS frame constructed by the system based on the example shown in Figure 7. The frame can now be used by the remainder of our system in the structural alignment phase.

7 Related Work

This work has parallels to MAIMRA, a system for word learning from non-linguistic input (Siskind, 1990). MAIMRA's semantic structure is also Jackendoff LCS, and its architecture consists of three modules: a parser (which produces syntactic parse trees from linguistic input strings), an inference component (which produces semantic structures from non-linguistic input), and a linker (which establishes correspondence between the syntactic and semantic structures). Observing that the parser, inference, and linker components respectively fill the linguistic processing, semantic processing, and structural alignment requirements outlined in Section 2, MAIMRA can be viewed as an instance of the general architecture we have described.

However, our system is also significantly different from MAIMRA in two important respects. First, MAIMRA is designed with the model-refinement aspect of word learning intertwined with the correspondence-inference aspect. In contrast, our architecture seeks to systematically isolate these two problems, so that problems of model refinement and correspondence establishment may be pursued independently. Second, MAIMRA's design results in exhaustive searches of many spaces (for example, the parser must generate all possible parses). Instead, our system seeks to use what we know as soon as possible, for example by using bidirectional search to guide the parse process. This implementation detail becomes important in practical applications because exhaustive searches of all possible parses severely limits the complexity of sentences that can be parsed.

The current work is part of larger initiative, the Bridge Project. Based on the work of the Genesis Group at MIT's Artificial Intelligence Lab, this project seeks to build cognitively complete systems—systems in which language, vision, motor, and other AI domains work cooperatively to achieve results which would have otherwise been unattainable.

8 Contributions

Effectively learning the meanings of words from non-linguistic input requires the development of representations and algorithms to determine correspondences between the linguistic and non-linguistic domains. Through this research, our contributions to this goal include:

- We propose a general architecture, based on structural alignment, for employing linguistic and non-linguistic context in word learning. The system bootstraps itself by using acquired words to learn new words. We define the necessary properties of semantic representations used in such a system. We also define the modules this system will require.
- We outline a system which implements this architecture for the specific semantic domain of vision. We identify LCS structures as an appropriate semantic representation, and we demonstrate techniques for extracting LCS from video. We also show a bidirectional approach to the parsing and alignment problem.

We currently have the components described in our implementation functional in isolation. The true merit of the system will be determined as we bring together all the pieces; thus our final contribution is the actual implementation of the systems described herein. It is our hope that our research will act as a springboard for the development of model refinement algorithms which have the advantage of support from semantic alignment systems such as ours.

References

John R. Bender. 2001. Connecting language and vision using a conceptual semantics. Master's thesis, Massachusetts Institute of Technology.

Lera Boroditsky. 2000. Metamorphic structuring: understanding time through spatial metaphors. *Cognition*, 75(1):1–28.

Bonnie Dorr. 1992. The use of lexical semantics in interlingual machine translation. *Machine Translation*, 7(3):135–193.

Dedre Gentner and Arthur B. Markman. 1997. Structure mapping in analogy and similarity. *American Psychologist*, 52(1):45–56.

Berthold Klaus Paul Horn. 1986. *Robot Vision*. McGraw-Hill, New York, New York.

Ray Jackendoff. 1983. *Semantics and Cognition*, volume 8 of *Current Studies in Linguistics Series*. MIT Press, Cambridge, Massachusetts.

Jeffrey Mark Siskind. 1990. Acquiring core meanings of words, representated as jackendoff-style conceptual structures, from correlated streams of linguistic and non-linguistic input. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics (ACL-1990)*.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, Massachusetts.

Shimon Ullman, 1996. *High Level Vision*, chapter 10, pages 317–358. Sequence Seeking and Counter Streams: A Model for Information Flow in the Visual Cortex. MIT Press, Cambridge, Massachusetts.

Deniz Yuret. 1999. Lexical attraction models of language. Submitted to *The Sixteenth National Conference on Artificial Intelligence*.