Intrinsic Representation: Bootstrapping Symbols From Experience

by

Stephen David Larson

Submitted to the Department of Electrical Engineering and Computer Science

in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2003

Copyright 2003 Stephen David Larson. All rights reserved.

The author hereby grants to M.I.T. permission to reproduce and distribute publicly paper and electronic copies of this thesis and to grant others the right to do so.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
JUL 2 0 2004
LIBRARIES

Author ....
Department of Electrical Engineering and Computer Science
August 22, 2003

Certified by ....
Patrick H. Winston
Ford Professor of Artificial Intelligence and Computer Science
Thesis Supervisor

Accepted by ....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

BARKER

Intrinsic Representation: Bootstrapping Symbols From Experience

by

Stephen David Larson

Submitted to the Department of Electrical Engineering and Computer Science
on August 22, 2003, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

If we are to understand human-level intelligence, we need to understand how meanings can be learned without explicit instruction. I take a step toward that understanding by focusing on the symbol-grounding problem, showing how symbols can emerge from a system that looks for regularity in the experiences of its visual and proprioceptive sensory systems. More specifically, my implemented system builds descriptions up from low-level perceptual information and, without supervision, discovers regularities in that information. Then, my system, with supervision, associates the regularity with symbolic tags. Experiments conducted with the implementation shows that it successfully learns symbols corresponding to blocks in a simple 2D blocks world, and learns to associate the position of its eye with the position of its arm.

In the course of this work, I take a new perspective on how to design knowledge representations, one that grapples with the internal semantics of systems, and I propose a model of an adaptive knowledge representation scheme that is intrinsic to the model and not parasitic on meanings captured in some external system, such as the head of a human investigator.

Thesis Supervisor: Patrick H. Winston
Title: Ford Professor of Artificial Intelligence and Computer Science

Acknowledgments

Above all, I owe this to my thesis advisor and mentor, Patrick Winston, for giving me the right conceptual tools and biases and allowing me the freedom to ride out to where only fools may tread.

I owe the idea of an interactive space, in addition to several other technical innovations of my system, to several insightful conversations with Karl Magdsick.

I owe significant conceptual clarity to Greg Detre, who has been extremely insightful in his comments on my drafts and has helped to make sure my arguments hold up to scrutiny.

I owe many ounces of clarity in my thesis to several insightful conversations with Raj Krishnan, Dr. Kimberle Koile, and Keith Bonawitz. Their comments on text I had written or ideas in my head were very helpful.

I owe my involvement in Artificial Intelligence to Jake Beal and Justin Schmidt. The former, who brought me on board the Genesis Project after being his student in 6.034 and helped guide me to my thesis topic, and the latter, who I aspired to follow as I watched him climb the ranks in this field.

I owe much thanks to all the individuals who have participated in the Genesis Project research group in the years that I have been at MIT. Conversations from this group have inspired me greatly.

I owe thanks to the other superlative and inspiring professors of artificial intelligence that I have learned from while at MIT, namely Rodney Brooks, Marvin Minsky, and Randall Davis.

And of course, I owe my ability to write this thesis at all to the love, nurturing, and respect of my parents, who have given me all the opportunities a human being could ask for.

Contents

List of Figures

List of Tables

Preface

I believe that the major hurdles in the path of achieving human-level artificial intelligence are conceptual rather than technical. I share the concern among some researchers that the pathway to achieving such a goal is still, after fifty years of AI research, hazy at best.

I feel that AI has an excellent opportunity to integrate knowledge from other brain sciences because of its emphasis on modeling theories with computers. Unfortunately, it is difficult to gather enough multi-disciplinary knowledge and instinct to be fluent in the languages of the other relevant sciences. Nevertheless, AI should reach out to these outside influences and use them to further restrict its design space. It needs to increase its focus on systems that are biologically inspired and neurologically plausible. It needs to focus on representations that brains make.

With this in mind, the hope of this thesis is to remove some of the obstacles along the pathways of integration between the brain sciences.

> *"Do not go where the path may lead. Instead go where there is no path and leave a trail."* – Gandhi

Chapter 1

Introduction

Imagine that it is the year 2053 and you are standing in front of a humanoid robot empowered with human-level artificial intelligence. This robot is lying on the floor in front of you, fresh out of the box. It has never been used. As you turn it on, you are surprised by the behavior that ensues. Rather than leaping up and doing intelligent things, the robot begins twitching on the floor. Its limbs jolt around randomly, and its eyes roam wildly. Believing that your product is defective, you take a moment to peruse the included instruction guide. You are once again surprised upon learning that this behavior is the normal start up process for the robot.

After a few minutes, the robot's motions become less erratic. Its eyes begin to focus on things in the world, and soon can keep its gaze fixed on objects as it moves. Its limbs, weak at first, begin experimenting with some toy objects included in the package. The robot picks them up and watches them fall.

As you continue to watch the robot, you observe it rapidly developing skills in the same way that human children develop them, only much faster. It begins walking around on unsteady legs, gradually gaining a confident walk. It inevitably makes some mistakes during this process, but learns as much from its mistakes as its successes. After a few hours, your robot begins prompting you to speak to it. After a few more hours of conversation, the robot is as capable with language as a 10-year old. It has essentially "grown up" before your eyes.

After observing this miracle, you begin to research the advances in the last fifty years of artificial intelligence that have enabled robots to have this kind of intelligence, and in particular why this kind of training process is necessary. You discover that soon after the turn of the century, AI research experienced a paradigm shift. In its early years, the field made a lot of advances solving hard problems using relatively simple representational schemes for storing information about the world. Only a few decades later, however, AI had run up against a wall in its attempts to create systems that could solve a broader class of problems. The old representational schemes were unable to draw much relevant information from the real world by themselves, instead relying mostly on human hand-coding of information to make sense of the world.

The representational schemes that ushered in the new age of AI focused less on information bundled into discrete elements. Instead, they focused on how information can be flexibly, quickly, and continuously collected from the world and how it can be profitably utilized to solve problems. Researchers discovered that by coordinating the learning processes of multiple sensory devices, their systems were able to learn about a lot of important relationships in the world. By organizing simple learned relationships in a powerful way, these systems were able to build new descriptions from old descriptions the way a human does. The units of representation, or symbols, that were generated had intrinsic meaning by virtue of the fact that they were created by a scheme already familiar to the system. Strangely, the designers themselves were not always able to fully interpret the descriptions of the world that their systems came to use internally, as they were grounded in complex nonlinear systems. This however, did not prevent them from being built, as the designers understood the principles that enabled the descriptions to be to be created and utilized effectively. More than anything, it was a complete understanding of these principles that enabled the creation of human-level artificial intelligence systems.

"Aha," you say to yourself, "this explains the learning processes I observed with the robot. It didn't come pre-packaged with all the data it needed to know, it was allowed to learn and build knowledge as it interacted with the world, the way humans do. It reminds me of the old parable, give a man a fish, he eats for a day, teach a man to fish, he eats for a lifetime."

1.1 Preview

The story in the previous section is a fanciful fictional account, but it outlines my vision of the end of a long journey for which this thesis is an early step. While we are still a long way off from building the kind of robot described above, I have designed and implemented a computer program that demonstrates some of the properties of the idealized system described above. Its key features are:

- Collects information from its environment quickly and flexibly in a way that can be used to solve problems.
- Discovers and organizes simple relationships in the information that it collects.
- Uses the results of its organizing process to bootstrap higher-level descriptions, such as symbols.
- Grounds the meaning of the descriptions it discovers in a semantics defined entirely by the system itself, thus creating *intrinsic* representation.

The system demonstrates these features by way of two experiments. In the first experiment the system *learns symbols corresponding to blocks in a simple blocks world.* In the second, the system *learns to associate the position of its eye with the position of its arm.*

1.2 Overview

In chapter 2, I give more specifics about the limitations of modern representations. I use the symbol grounding problem as a motivating problem, discuss the nature of representation, and introduce the ideas of semantic completeness and knowledge generation. I turn to a more detailed explanation of the model of intrinsic representation in chapter 3. In chapter 4, I describe the significant components of my implementation. In Chapter 5, I tie the elements introduced in chapter 4 together and explain the architecture and implementation of my model. In chapter 6, I analyze the implications of my work in light of the issues brought up earlier. In chapter 7, I summarize the contributions I have made.

Many influences prompted me to write this thesis. In part, I was influenced by a broad set of readings across several different fields. To provide a sense of these fields, I have provided a multi-disciplinary review in Appendix A. Additionally, I compare and contrast other work similar in spirit to my thesis, by, for example, Agre and Chapman, Drescher, and Roy, in Appendix B.

Chapter 2

Representation

"Once a problem is described using an appropriate representation, the problem is almost solved." (Winston 1993)

How the brain represents the building blocks of thought is one of the remaining unsolved mysteries of human existence. That the brain creates representations of some kind is clear; we observe many organisms acting on prior information. Such behavior necessitates that organisms appropriately store and retrieve information. In humans, these representations are well suited to enable the performance of an endless number of tasks. What is less clear is how the brain represents this information in such an advantageous way.

Because computer models are dynamic and can be updated quickly and on demand, sophisticated computer models have been made possible. Today, computers are used for modeling a variety of complicated things–car engines, economic theories, and the human genome are just a few. Our understanding of the representations used in the building of models has improved in proportion to our understanding of how to build models at all. What we have discovered is that within narrow, well-defined problem domains, finding the right representations is a matter of reductionism. Within these domains, good representations are found first by understanding the basic principles of the desired application, and then by matching an appropriate encoding scheme. However, in problem domains where the basic principles are unclear, finding good representations is much more difficult. How the brain builds representations that enable it to carry out tasks for survival is currently one of those uncharted domains. Unfortunately, our ability to make *some* powerful representations has not yet translated to an ability to make the kind of powerful representations the brain can.

2.1 The Symbol Grounding Problem

One of the drawbacks of traditional symbolic AI reasoning systems is their domain specificity. What causes this? Harnad, with his description of the symbol grounding problem, sheds some light on this question. He describes the problem as the following:

> "How can the semantic interpretation of a formal symbol system be made intrinsic to the system, rather than just parasitic on the meanings in our heads?" (Harnad 1990)

A symbol is merely a label for a larger body of knowledge that is difficult to describe without that label. Harnad says that the label should at no point be confused for the larger body of knowledge to which it refers.

An example of this problem is seen by considering a person reading the word "chair." This action summons a mental image of the object. However, the perception of the printed word by itself is not equivalent to the perception of the mental image. One can think at length about a chair beyond the five letters of its English representation. One can consider its various aspects, chairs that one is familiar with, and so on. Those abilities must be fueled by information that is obviously not contained in the letters C-H-A-I-R. The letters are more like a trigger. Our extensive ability to recall information about a chair makes it clear that there is more to our mental representations of words than their printed image.

For a symbolic system, however, the printed image is almost exclusively what represents an object. To date, symbolic systems lack the ability to intrinsically interpret the meaning of data that is intended to represent a chair. For example, some of the most common symbolic representations of traditional AI are semantic nets and frames. These representations describe a data object as a collection of other data objects that model its different properties. For example, a chair may be a member of the category of 'furniture' and may have the texture of 'wooden'. However, these associated properties of a chair are merely other symbols; other labels that humans give meaning to.

This limitation does not prevent us from creating systems that function with competency at specific tasks such as playing chess. This is because within the domain of chess, we are able to define all the relevant information about the symbols. The legal moves of a chess piece are easy to program and do not ever vary. However, when we attempt to create systems to operate outside of a limited domain, their functionality is limited by the information that can be given to it, as well as the way that information can be stored.

The symbol grounding problem challenges us to distinguish between two separate concepts. On the one hand, we have a symbol as analogous to markings on a piece of paper. On the other hand we have the mental processes that allow us to consider a symbol's meaning. It also forces us to consider that our brains store more information about things in our world than we are able to describe.

For example, when we look at a toy block, we have a wealth of information about it at our fingertips. We can produce language concerning the block; we can say its name, its color, its shape, or describe its purpose. We also can solve problems with it, such as figuring out if it would fit inside a particular box, or if it could be used to hold up other blocks. We know what it feels like to pick it up, and we know what happens if we balance it on its corner.

As you read the previous paragraph, you recall these different aspects of a block for yourself. By invoking the symbol "block", you have triggered your personal knowledge about a physical object. Is the description in your brain equivalent to the description given above? Are there really English sentences stored in your brain that *tell* you what a block feels like in *words*? Or is it possible that it is represented in a separate way that *enables* a description in language? Most likely we do not use only verbal descriptions of things to store information about familiar objects, but rather, verbal descriptions provide a convenient way of aggregating that information.

As a solution to the symbol grounding problem, Harnad proposes to under-gird the computer descriptions of symbols with non-symbolic representations derived from sensory experience. Such non-symbolic representations are real-valued, and are derived from the information received from our most peripheral neurons. The meaning of symbols can be defined as the recollection of experiences associated with them if sensory experience is used as the greatest common denominator representation. For example, a toy block can be represented as the combination of all of the sensory and motor interactions that one has had with a toy block.

Clues from the field of neuroscience suggest that some form of this kind of representation is found in the brain.

> "Recent functional brain imaging studies suggest that object concepts may be represented, in part, by distributed networks of discrete cortical regions that parallel the organization of sensory and motor systems."(Martin and Chao 2001)

We might imagine that a symbol, such as the printed word form of an object, acts as an index that activates appropriate combinations of cortical brain systems. The "meaning" of a printed word symbol would then come from the process of distributed cortical systems contributing information about aspects of the object the word symbol refers to. With the ability to represent symbols as combinations of sensory experiences, we may also achieve a deeper understanding of how the human brain can so profitably integrate information from different sensory modalities.

For AI research, the symbol grounding problem is an obstacle to creating systems that are capable of more human-like knowledge representations. Unfortunately, we still do not have a clear idea of what qualities we should aim to give symbols in order to make them more grounded. Below I suggest that what is needed is a more rigorous and general definition of representation.

2.2 Defining Representation

The following section is philosophical in nature, and represents an important contribution of the thesis. There are many ways to arrange a thesis, and my way is to explain the personal thoughts that lead me to my work before explaining my solutions. The less philosophically inclined reader, however, may want to forgo the bulk of this discussion and focus only on section 2.2.4 on the first read through, as it provides some definitions that are used later on.

(Winston 1993) provides us with a useful definition of representation:

> "In general, a representation is a set of conventions about how to describe a class of things. A description makes use of the conventions of a representation to describe some particular thing."

He goes on to explain that a representation has four fundamental elements, consisting of, quote:

- A lexical part that determines which symbols are allowed in the representation's vocabulary
- A structural part that describes constraints on how the symbols can be arranged.
- A procedural part that specifies access procedures that enable you to create descriptions, to modify them, and to answer questions using them.
- A semantic part that establishes a way of associating meaning with these descriptions.

This view of representation is heavily influenced by computer science. One sign of this is the usage of *symbols*, which are thought of as atomic units of representation and are easily represented in computers. Another sign is that its description of procedures operating on descriptions sounds akin to the functioning of a computer system.

All this is as it should be. Artificial intelligence is a field rooted in computer science, and data representations are a crucial tool in the design of computer systems.

However, is this the most useful definition of representation to use when analyzing the representational abilities of the brain, a long-standing goal of AI? I will argue below that this definition must be expanded in order to allow for systems to be designed that represent in a more powerful way.

2.2.1 The Meaning of Semantics

(Winston 1993) continues on to discuss the meaning of semantics. He mentions three schools of thought that concern themselves with this question. Quote:

- Equivalence Semantics. Let there be some way of relating descriptions in the representation to descriptions in some other representation that already has an accepted semantics.
- Procedural Semantics. Let there be a set of programs that operate on descriptions in the representation. Say that meaning is defined by what the programs do.
- Descriptive Semantics. Let there be explanations of what descriptions mean in terms we understand intuitively.

He feels that ultimately we always end up back at descriptive semantics:

> " In the case of equivalence semantics, descriptions have meaning because they are equivalent to something that means something to you. In the case of procedural semantics, descriptions have meaning because they cause a program to exhibit a behavior that means something to you. Thus, the alternatives all seem rooted in perceptions to which you ascribe meaning intuitively."

What if there is more to this story? What if there are descriptions that have meaning in a way that does not depend on their translation into something intuitively meaningful to us? What I wish to talk about is an *internal semantics*; a semantics that is meaningful to a system without necessarily being meaningful to us.

Why should you care about such a thing? Were you to build a system whose internal semantics were not meaningful to you, you would likely think it was broken! However, we encounter systems that have unknown internal semantics all the time. They are called brains. While man-made systems require a known semantics, natural systems are not always as convenient. Sometimes we care about reverse-engineering, and consequently should be concerned about the existence of an internal semantics that we do not yet fully understand.

More importantly, however, is the point that descriptive semantics avoids an extremely important question: what does it mean to have an intuitive understanding? Descriptive semantics seems to suggest that there is no simpler way to understand semantics beyond intuition, but does not explain what it means for our brains to do so. For this school of thought, the problem of the homunculus rears its ugly head.

Because brains are themselves systems, it seems that an explanation of the internal semantics of systems would go a long way towards finding the foundation upon which these intuitive understandings are built.

2.2.2 Internal Semantics

Let's talk about what makes a semantics meaningful to a system. What is a system? A group of interacting, interrelated, or interdependent elements forming a complex whole. What is a semantics? For our purposes, the best definition is a set of constraints that specify how the state of a system changes. (Bickhard and Terveen 1995) suggests that as the state of a system changes, the meaning of its semantics is *the effect to the system of propagating its constraints in response to those changes.* In a nutshell, semantics and dynamics are intertwined.

Let us make Bickhard's idea more concrete with an example. As you read text, the information from the page causes your brain to change its state. The internal semantics of the system that is your brain are the complicated dynamics of neuronal interaction. Those dynamics impose constraints on the way that your brain's state changes in response to the text. What makes the text mean something to your brain is the relationship between the state changes brought about by 1) the text physically changing your retina's neurons and 2) the state changes that propagate as a result. What kind of changes propagate? For our brains, it is those state changes that enable us to do all the things that we traditionally understand as having gathered the meaning of something. Those resulting state changes allow us to act on the new information in an appropriate way.

The beauty of this understanding of semantics and systems is that it *generalizes across systems.* Let us take the example of a simple biological system: an amoeba. As a single, one-celled creature, an amoeba has no brain. It is, however, a complex system made up of interacting molecules. When exposed to a sugar gradient, the amoeba will travel up the gradient to take advantage of the better food source. Enabling this movement to occur are molecular processes capable of detecting sugar levels and translating that detection into motion. The internal semantics of the amoeba system are defined by the constraints imposed by its molecular processes. Under this definition, the propagation of constraints that the sugar molecules enable is the meaning of the sugar molecules to the amoeba. Note that this meaning is completely defined by its internal dynamics, and without complicated computer simulations rather unintuitive to us.

Let us examine a system familiar to artificial intelligence: a chess-playing system. We are familiar enough with its representations: a board and pieces that occupy positions on the board. State changes are made by changing the locations of pieces on the board representation. What makes those changes meaningful to the system? The internal semantics of the chess-playing system are defined by the data relationships provided for by its code. As the result of that code, when we change the state of the chess-board, certain preconditions may now be met that were not previously. Actions that were only potential actions before can now be enacted. Again, it is the relationship between state changes and their dynamic consequences on the system, defined by its constraints, that gives meaning to the changes. Keep in mind that this is a totally separate issue from the meanings that *you* would assign to the state changes made by the chess player. Those meanings would not come from the chess-playing system, but rather a different system entirely: your brain.

2.2.3 Interactive Spaces

So far I have discussed how the idea of internal semantics can be applied to very different systems such as the brain, an amoeba, and a chess-playing computer. While we can describe the *existence* of internal semantics in systems, I maintain that we can only *understand* them by making models that approximate their functionality. While these models can be very useful, they should not be confused with a fully "objective" understanding of a system. This is because each model invariably inserts some bias, such as the choice of granularity of the description. Nonetheless, we are interested in the methods of creating models of the internal semantics of a system. In order to demonstrate one way this can be done, I propose an abstract representation for the internal semantics of a system: an *interactive space.*

The interactive space of a system is a high-dimensional mathematical space. A point in this space defines a system's activity in response to a state change. If we imagine a state-transition diagram where the states are nodes, and the transitions are links, the points of the interactive space of that system would be equivalent to the links. Note that state changes can be prompted both by the environment and by the system itself. The axes of this space describe a set of continuous variables which define the states the system can be in.

In the interactive space of the chess system, a point would be equivalent to the dynamics of a transition between two board states. In the interactive space of the amoeba, a point would be equivalent to the dynamics of a transition between two states of molecular dynamics. In the interactive space of the brain, a point would be equivalent to the dynamics of a transition between two states of neuronal dynamics. The definition of an interactive space can vary in the granularity of its axes depending on what level of abstraction you wish to use. You could define the axes of the chess program in terms of the states of the processor it is running on or in terms of the language it is written in. You could define the axes of the amoeba in terms of proteins, molecules, atoms, quantum particles, etc. All these levels apply to the brain as well, as does the neuronal level. Although these spaces may seem unwieldy and only conceivable in theory, our ability to define them even in an abstract way gives us more investigative power over them than we had without them.

Consider that subregions of interactive space correspond to behaviors that humans would give intuitive meanings if observed. Some subregion of the interactive space of a chess program corresponds to the program putting its opponent in check, while some other subregion corresponds to the program castling. If the program is particularly skilled, we might be able to call a subregion of its interactive space the program responding to pressure from its opponent. For the amoeba, we would likely be able to define some subregions of its interactive space as kinds of behavior. Swimming up a sugar gradient is itself an example of a subregion. Reproducing might be another. Finally, this notion can be extrapolated to the brain. The range of human actions, skills, and behaviors could all inhabit their own subregion of the brain's interactive space.

It is worth mentioning that an interactive space is fully contained inside of a meta-interactive universe. While the interactive space is constrained by the dynamics of its system, the universe is not. The points that lie outside of the interactive space of a system correspond to violations of its dynamics. The meta-interactive universe is essentially a bizzaro-world. Points outside of the interactive space of a chess system correspond to pawns that move like queens, or multiple simultaneous moves by pieces from both sides. Points outside of the interactive space of an amoeba correspond to the amoeba suddenly transporting from one side of a petri dish to another. Points outside the interactive space of the brain correspond to telepathy and telekinesis. Why is it useful to talk about the bizzaro-world outside of the constraints of a system? Because by doing so, we help to clarify the fact that the constraints of a system do create boundaries, and this helps us to understand where those boundaries lie.

2.2.4 Implicit and Explicit Representation

An interactive space is an abstract example of an *implicit representation*. An implicit representation is a set of conventions for describing things in a way that is unformalized. It is a scruffy way of capturing information that does not have a one-to-one correspondence between its data and an intuitive understanding of it. It represents information that is subject to the eye of the beholder. Descriptions that result from an implicit representation generally look like a large lump of data. An example is a real-valued read out of all the voltages in all the neurons in your head. Another example is a real-valued read out of all the x, y, and z coordinates of every point on your body. In both cases, we would have to apply interpretations to the description in order to turn them into data we could use. In both cases, we could slice that data up in many different ways to solve many different problems.

By incorporating this formalism, we can define *explicit representations* as a neat way of capturing information that generally *has* a one-to-one correspondence between its data and an intuitive understanding of it. Generally, explicit representations are designed to make each description have only a single interpretation. Frames, search trees, semantic nets, and all other representations that follow Winston's definition of a representation above fall into this category. Now that we have defined interactive spaces, we can expand our notion of explicit representation to include internal semantics. An explicit representation of the internal semantics of a system can be formed by defining its interactive space and labeling its subregions, thereby creating only one-to-one correspondences.

Engineers prefer explicit representations for systems they construct. Because of this, we are accustomed to having our explicit representations describe the internal semantics of our man-made systems as concisely as possible without sacrificing what is deemed to be the most relevant information. Good engineering practice is to not build things that have a set of dynamics you do not fully understand. Man-made systems put effort into making their dynamics as predictable as possible. Major engineering catastrophes occur when unexpected dynamic effects manifest themselves.

On the flip-side, sometimes we can construct systems that do what we want, but for which we have a more difficult time creating explicit representations that explain why. The following case studies examine two examples that fit this profile.

2.2.5 Case Studies: Creatures and Neural Nets

In order to solidify the connection between internal semantics and implicit representations, I want to look at two case studies.

Brooks's Creatures (Brooks 1991) claims to describe a way to create systems that are intelligent without using any representations. One section of this paper is extremely relevant:

> "Even at a local level, we do not have traditional AI representations. We never use tokens which have any semantics that can be attached to them. The best that can be said in our implementation is that one number is passed from a process to another. But it is only by looking at the state of both the first and second processes that a number can be given any interpretation at all. An extremist might say that we really do have representations, but that they are just implicit. With an appropriate mapping of the complete system and its state to another domain, we could define a representation that these numbers and topological connections between processes somehow encode. However we are not happy with calling such things a representation. They differ from standard representations in too many ways. There are no variables [...] which need to be selected through pattern matching. There are no choices to be made."

Under the conventions that have been described, I would say that Brooks is both right and wrong. I maintain first and foremost that Brooks's Creatures have an internal semantics. His description of "the numbers and topological connections between processes" is precisely what I mean by an internal semantics. He is right to say that an implicit representation such as a mapping could be imposed on top of these semantics[1], and he is right to say that he is not obligated to do so. But I think Brooks misses a big opportunity when he essentially says that examining representation is a waste of time. By studying the internal semantics of the system and imposing some

[1]This sounds like an interactive space to me...

useful implicit representations on top of them, we are only more likely to reach a better understanding of why such systems can do what they do. When we can turn those implicit representations into explicit representations, we are only more likely to increase our ability to engineer and reverse-engineer those systems.

Interactive spaces give us a way of conceptualizing how to turn the internal semantics of a system into explicit representations. Here's how such a strategy would be laid out:

1. Increase understanding of the internal semantics of the system. For example, this could mean improving our understanding of Creatures as a dynamical system.
2. Use what we know about the internal semantics of the system to describe an interactive space. Using the language of dynamics, it may be possible to define a comprehensive set of variables that explain the behavior of the Creatures. Snapshots of these variables could be defined as states, and their transitions as points in the space.
3. Define an interpretation of that space that is as descriptive and simple as possible. This may mean making the right choices of how to label subregions.
4. Model this interpretation using a computer.

Perhaps by following this methodology, a greater understanding of subsumption architectures could be reached that would enable us to scale them up effectively.

Neural Networks The second case study looks at the construction of implicit representations by neural networks. (Robinson 1992) suggests that one of the reasons that engineers have been uncomfortable using neural networks to design systems is their inability to analyze the representations learned by their hidden layers. A simple feed-forward network that maps images of handwritten digits to the numbers zero through nine will produce hidden layers that are difficult to interpret. As Robinson says, "Hidden units contain bits and scraps of signals that yield only arcane hints about network function and no information about how its individual units process signals." He goes on to say that rather that think about a how a network represents as the sum of how its units represent, we may have to be content with mathematical descriptions that put the entire system in a black box.

The internal semantics of a neural network are determined by the dynamics of unit activation and the dynamics of unit interconnectivity. To translate Robinson's message in terms we are now familiar with, he is saying that we may need to rely on the more convenient implicit representation of the dynamics of the network rather than the less convenient implicit representation of the states of its hidden units. Both are ways of explaining the internal semantics of the system of a neural network. The hidden unit representation currently does not allow for the creation of a good explicit representation, while the dynamic one does.

These case studies have solidified the notions of internal semantics, implicit and explicit representations. Additionally, it was the goal of this section to define representation in a broader manner that allows us to view the representational abilities of systems that are not digital computers. Now that we have expanded our definition of representation, let us examine some additional characteristics we should want from representations in intelligent systems.

2.3 Additional Desiderata

An important tool for the measuring of progress is the use of goals. In the study of representation, it makes sense to define functionality goals, or in other words, what we expect that better representations should be able to accomplish that current ones cannot. The following two sections describe goals that we should look for in a system claiming to make an advancement in the design of representations for intelligent systems.

2.3.1 Semantic Completeness

Semantic breadth is defined as the ability for a representation to capture knowledge about a broad range of topics. It can be used as a comparative property of representations—representations can be more or less semantically broad than one another. Consider an example of abstraction in concepts. The concepts wooden-thing, block, and rectilinear-block are ordered from more general to more specific, and from more abstract to less abstract. We would call wooden-thing more semantically broad than rectilinear-block, because the former captures knowledge about a broader range of topics. Blocks are not the only wooden-things, so are trees and chairs.

The corollary to semantic breadth is semantic depth. Semantic depth is defined as the ability for a representation to capture knowledge at a fine resolution. It can also be a comparative property. With the wooden-thing example, rectilinear-block is more semantically deep than wooden-thing because it gives us more detail about the type and shape of the wooden-thing.

In the wooden-thing example, semantic breadth and semantic depth are inversely related. What we would like is to achieve representations that are the best of both worlds—both broad and deep. Such idealized representations we will refer to as *semantically complete.* Our target for semantically complete representations will be those that must be in brain that allow us to know about both a broad range of topics and at the same time be able to understand a great deal of detail about each of them.

The symbol grounding problem points out why symbol systems are less semantically complete than brains—because as discussed above, symbols usually mean more to brains than they do to computers. Symbolic systems have limited semantic breadth because they are trapped into narrow domains. They have limited semantic depth because we are unable to give computers as much detailed information about things as we are able to collect about them ourselves.

2.3.2 Knowledge Generation

While some AI systems are capable of generating previously unknown information, no system yet can give us back significantly more information than we put in. Human beings can do this.. Human beings have the ability to generate knowledge on top of knowledge in a cycle that appears limitless. It appears that our ratio of information out to information in is very high. The question is, how can we create systems that share this capability?

Modern AI's weakness is solving problems where symbols are unavailable or outside of the existing "symbol realm." Systems capable of knowledge generation would be able to fill in the gaps in its knowledge. A desirable system along these lines would be able to identify information that it lacks, and do something to acquire, represent, and use that knowledge effectively.

In section 2.2.4, I discussed the idea of implicit representation. I alluded that some systems that use implicit representations must do so indirectly. In the case of the hidden layer of the neural network, the descriptions that are formed are not designed by a person, but rather, generated as the result of a system being trained. As we delve further into systems that create difficult-to-analyze descriptions, we will likely have to rely more on systems that generate their own knowledge, because we will be unable to create it ourselves.

Chapter 3

Model of Intrinsic Representation

The model of intrinsic representation is designed to enable the association of symbols with their descriptions. The descriptions of symbols are discovered from the regularity inherent in the world. Its key differences from traditional symbolic systems are:

1. Symbols' descriptions are discovered from the statistical processing of experience.

2. Symbols' descriptions are equivalent to statistical regularities found in information spaces.

3. Symbols' descriptions carry their context with them by being situated in information spaces.

3.1 The Model

Figure 3-1 is the key diagram for the understanding of the model of intrinsic representation. At the bottom of the diagram, sensor arrays receive streams of data from the outside world. Such arrays could be imagined as a patch of skin or light sensitive cells in the retina. As data comes into the system through these sensors, it travels to a subsystem devoted to organizing and storing the regularities in the data. This subsystem arranges these regularities with respect to their similarity, placing highly similar regularities near each other, and dissimilar regularities farther apart. After a critical period, clusters of high similarity group the regularities into sets. At this point, data in the incoming stream can be treated as a trigger for the activation of the cluster of which it is a member. As clusters are activated by incoming data, they are associated together by their frequency of coincidence. The more often two clusters are active simultaneously, the more associated they will become. The resulting trained system treats incoming patterns as members of a class, and can react to the activation of that class[1].

3.2 Theoretical Foundation

This model is built on top of a few theories about the way information works in brains. First is the theory that there is enough statistical regularity in the environment from which brains can reasonably form useful symbols. Second is the notion of an information space as a definable entity in either brains or computers. Finally, we revisit the notion of implicit representation defined in section 2.2 and analyze the degree to which the model uses it. Let us cover each theory in turn.

3.2.1 Regularity in the Environment

The information brains collect about the environment contains statistically detectable patterns. There are several reasons to believe that this claim is true.

From a systems perspective, brains must reduce the informational entropy coming from its primary sensory areas. Psychologists have commented that one of the

[1]While figure 3-1 illustrates only two information spaces associating together, the model is consistent for any number of information spaces. If we were able to identify all of the information spaces that the brain gets input from, this model should be able to create associations between all of them. The idea behind this aspect of the model is to mirror a theory of brain evolution. The brain may have evolved to create more maps that are pointed at information spaces not previously being learned in order to increase chances of survival. For example, a primitive visual system might not have had color receptors. However, color information is useful to animals, so the information space of visual input expanded to contain color information. Later on, there may have been need for more information processing. A higher order visual information space, such as shape information, could have been generated from the lower-level information space of visual input. This information could be integrated with other sensory information coming from the ears, the skin, etc. As information became useful, it may have been an evolutionary advantage to direct a mechanism at it that was capable of extracting regularities from it.

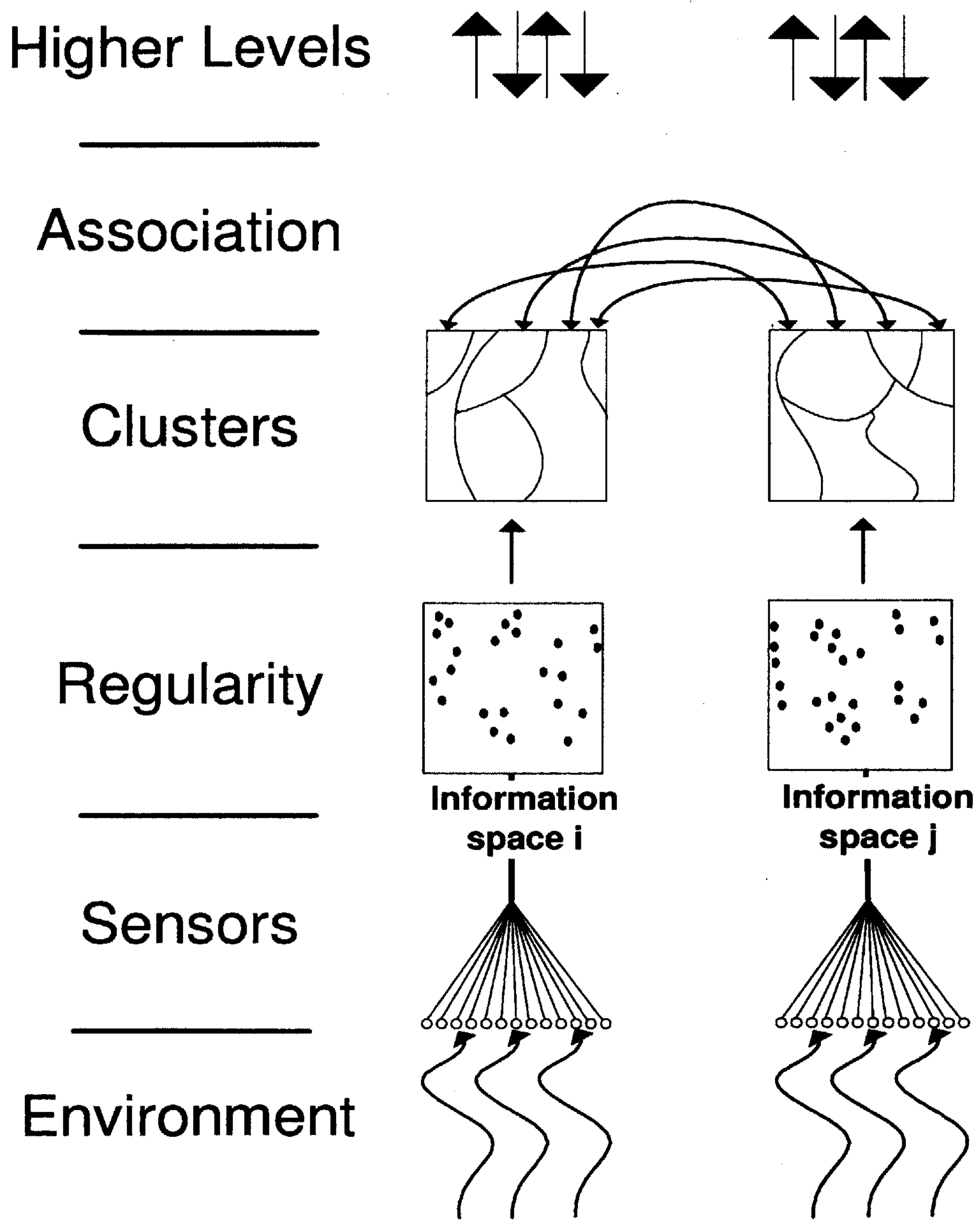

Figure 3-1: The model of intrinsic representation. Two information spaces, i and j, are shown side by side. At the bottom, information from the environment impinges on the most peripheral sensory neurons, creating signals in information spaces. Over time, the regularity in the information space is stored in a map by way of a process of self-organization. From this map, clusters of high similarity can be segmented and reified as their own units. Associations are created and strengthened based on the co-activation between clusters in different information spaces, creating networks of activation. As members of these networks, clusters can be used as inputs in new information spaces and can be associated with other co-occurring units in different systems, allowing reactivation of local clusters from afar.

surprises about the brain is not that it collects so much information, but rather that it manages to filter out irrelevant information. Without this ability, we would never be able to focus on anything, and we would lose the survival game. Thus, in order to make sense of the world, the brain must find relationships in the information it receives.

The brain has some help. While the amount of information coming from the world is immense, it could be much worse. A significant amount of regularity exists in environments that brains have evolved to interact with. We do not live in a world of white noise. We live in a world where matter has permanence, and tends to be localized. Laws of physics are consistent. Thus there are certain classes of information that are prevented by our environments from ever reaching us. The information that I have stepped on a rock and accidentally fallen *upwards* is an example of something that evolving brains have not had to compensate for. The regularity in our environments gives us a leg up when attempting to make sense of the world.

3.2.2 Information Spaces

I introduce the idea of an information space as a useful way of conceptualizing streams of information in brains. The best physical metaphor for an information space is the data traveling along a bundle of wires. A point in the space is an n-dimensional vector capturing the values of each wire at a given instant in time. Information spaces are as vector spaces applied to the brain, and all of the language of Linear Algebra is applicable in analyzing them.

For example, let us think about the information flowing from the optic nerve into V1. If we were to take a cross section of the optic nerve just before it enters V1, we could analyze the dynamics of all the neurons that compose it at that point. If we imagine each neuron as a wire carrying a voltage, at any given time, each will have some real-valued level of excitation. As data flows through this cross section, we can imagine that a point travels through some kind of trajectory through this n-dimensional space. All possible points in this n-dimensional space comprise the information space.

Information spaces can be defined arbitrarily, so long as you can select a group of wires. You can define an information space as the information flowing along neurons leaving V1 entering V2. You can define an information space as the information flowing along neurons inside a particular part of V1 at a particular place. The trick is to identify the spaces that are useful.

At its root, an information space is defined by 1) its dimensionality, and 2) its features[2] The information space leading into V1 is very different than the information space leading into the primary sensory areas. Different things will be salient to different spaces.

3.2.3 Implicit Representation

After the discussion of implicit representation in section 2.2.4, it is appropriate to ask about the relationship between implicit representation and intrinsic representation. First of all, intrinsic representation seeks to understand how explicit representation can arise from systems that use implicit representation internally. Discussions about symbols, therefore, key into our notions of explicit representation. Intrinsic representation seeks to generate explicit symbols from implicit representational systems.

Additionally, intrinsic representation focuses on the fact that symbols generated from implicit representational systems have an inherent meaning to that system, while symbols designed purely explicitly are only meaningful insofar as they can be interpreted. The *intrinsic* nature of these representations relates to the fact that they carry their meaning along with them, rather than having their meaning given to them by another system.

3.3 Key Points

As described above, the key differences between the model of intrinsic representation and traditional symbol systems are:

[2]A peek ahead to figure 5-8. illustrates what these features are in my implementation.

Symbols' descriptions are discovered from the statistical processing of experience. In almost every traditional symbol system, the symbols *and* their descriptions must be provided before-hand. In the model of intrinsic representation, the descriptions of symbols are discovered by processing sensory data in an unsupervised manner.

Symbols' descriptions are equivalent to statistical regularities found in information spaces. The nature of the symbols in traditional symbol systems are as tags whose descriptions are provided by a human designer. Descriptions formed from statistical regularities do not have to be provided by a human designer. As a result, symbols will represent certain classes of things in the world not because a designer finds them useful, but because those things are the most statistically salient in the given information space.

Symbols' descriptions carry their context with them by being situated in information spaces. Traditional symbol systems require context as an extra parameter to make sense of a symbol. Symbols may mean different things in different contexts. This is because a symbol is thought to be a separable part of a system rather than an intrinsic part of a system. With intrinsic representation, however, because symbols are derived from information spaces, they are inextricably linked to those information spaces, and carry no individual meaning outside of them. This is because a symbol is associated with a cluster, which in turn defines a set of vectors that are defined in terms of the dimensionality and features of the information space[3]. A cluster in an information space of visual information cannot be transported into an information space of any other kind of information. As a result, you cannot have a symbol without its context.

[3]A glance forward at figure 5-8 shows the features of the vectors that I am using, which in turn define the information space.

Chapter 4

Tools for Implementation

This chapter sets up chapter 5 by introducing the significant algorithms and technologies used in the implementation.

4.1 Self-Organizing Maps

The Self-organizing map (SOM) algorithm performs unsupervised learning on a set of incoming n-dimensional input vectors. In its basic form, it is visualized as a sheet-like two-dimensional array of cells. As the SOM is trained, the cells become tuned to patterns in the input. The resulting map reflects an organization that can be thought of as a 2D projection of the most salient relationships in n-dimensional space.

Self-organizing maps stem from a line of research into statistical learning algorithms that include methods like vector quantization and principal components analysis. Self-organizing maps are an innovation on traditional statistical learning algorithms due to the multidisciplinary influences that guided its creation. Self-organizing maps try to mimic some of the functionality reported in the sensory feature maps of the cortex. While it is sometimes called a neural networks algorithm, its cells do not function in a manner that is consistent with standard models of neurons. SOMs are designed to be a biologically plausible system that is built at a level of abstraction above that of the neuron level. The idea of modeling the activity of a group of neurons all at once in a high-level, easy to implement algorithm is one of the reasons that SOMs are so popular. (Kohonen 2001) has a complete survey of the applications of the SOMs idea that have been created in the short time since the algorithm's discovery.

My conception of the three important purposes of self-organizing maps to artificial intelligence research are:

1. SOMs as an adaptive representational system.

   The self-organizing map arranges data by performing an incremental statistical analysis on the information it receives. As each new piece of data arrives, it is fit into an appropriate place in the landscape of existing data. Novel data that differs significantly from existing data eventually carves out its own space on the self-organizing map. This allows the self-organizing map to aggregate information flexibly.

2. SOMs as a content-addressable memory.

   Each cell in a self-organizing map serves as a record of data that it has been exposed to. Within the array of other cells, each cell describes a particular class in the landscape of data represented by the map. Novel data can be matched to its most similar cell on the map, and thus classified by way of its position with respect to other cells. In this way, a SOM allows data to be stored and retrieved using its content as its address.

3. SOMs as an implicit representational system.

   A SOM is a curious thing by virtue of the fact that it is an excellent hybrid of explicit and implicit representational systems. On the one hand, a SOM creates a visual representation of a large body of data, which allows relationships in that data to become obvious to a human viewer. On the other hand, a SOM does not make the relationships in the data explicit. It is left up to the observer to derive explicit rules and relationships for themselves. Fortunately, explicit representations are not difficult to draw out. In particular, one can derive clusters from a SOM that allow the grouping of high-dimensional data into classes. One can discover which clusters are near one another, and which are far away. As a result, SOMs are an excellent candidate for the job of combining implicit and explicit representations.

4.1.1 The Basic Algorithm

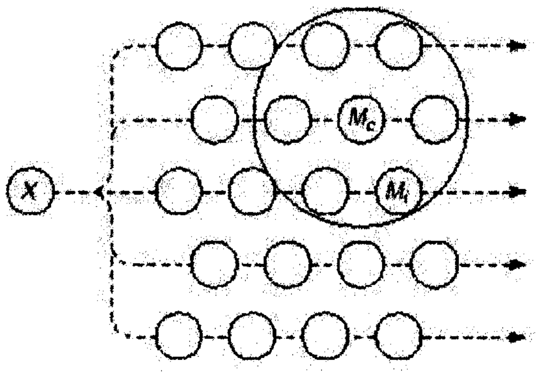

Figure 4-1: A self-organizing model set. An input message X is broadcast to a set of models M_i, of which M_c best matches X. All models that lie in the vicinity of M_c (larger circle) improve their matching with X. Note that M_c differs from one message to another. [from fig.1 in (Kohonen and Hari 1999)]

Here is how the basic algorithm works. Each training iteration begins with the selection of an input pattern, $x(t) \in \Re^n$, where t is a discrete-time coordinate. Each cell i in the map has a model vector, $m_i(t) \in \Re^n$ associated with it. After initializing all cells in the map to a random distribution:

1. Find the winner cell $c(t)$ whose model vector, $M_c(t)$, best matches $x(t)$.

$$c(t) = argmin_i \| \ x(t) - m_i(t) \ \| \tag{4.1}$$

   In the basic model, Euclidean distance is used as a similarity measure.

2. Discover neighborhood $N_c(t)$, which are the cells surrounding the winner cell c

3. For each $i \in N_c(t)$, update the model vector as follows:

$$m_i(t+1) = m_i(t) + \alpha(t)[x(t) - m_i(t)] \tag{4.2}$$

4. Otherwise, the model vector stays the same:

$$m_i(t+1) = m_i(t) \tag{4.3}$$

This process repeats for all the patterns available.

$\alpha(t) \in [0,1]$ refers to the learning rate at time t. Both α and $N_c(t)$ are typically reduced as the learning process evolves to allow for more focused representations towards the end of the training session.

4.2 Growing Self Organizing Maps

Growing Self Organizing Maps are an extension to the basic SOM idea that allows maps to increase their quantity of cells over time. It is described in detail by in (Dittenbach, Merkl, and Rauber 2000). That reference actually defines the notion of a hierarchical growing self organizing map, but only the growing aspects of the map were used in my implementation. Let us discuss the extra machinery that allows maps to grow.

Whereas in the basic SOM model, maps have a set number of cells that does not change throughout the training phase, GSOMs begin with a 2x2 map which grows over the course of training.

The most important reason to use GSOMs is that the choice of a fixed topology with the basic SOM is a design decision that is difficult to justify. Allowing that decision to be made by the statistical structure of the data leaves one less thing to worry about. Additionally, the training process of a GSOM goes much more smoothly because cells are grown into appropriate places in the map, rather than being initialized randomly.

4.2.1 Map Growth

The basic SOM algorithm is carried out on the map to begin. Every λ iterations of the algorithm, a growth phase is started. During the growth phase, an error cell, e,

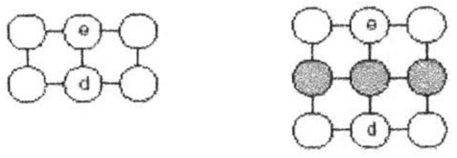

Figure 4-2: The insertion of cells [from (Dittenbach, Merkl and Rauber 2000)]

is selected. Next, the most dissimilar neighboring cell, d is selected. Finally, a new row or column of cells is created between e and its most dissimilar neighbor d.

The selection of e involves the computation of a value known as the quantization error (qe) for each cell. Here is the formula for qe:

$$qe_i = \sum_{x_j} \parallel m_i - x_j \parallel \tag{4.4}$$

For each cell i, we compute the similarity between every input vector x_j that is mapped onto cell i and compute the sum of the difference between the model vector m_i and x_j.

The error unit e is the cell with the highest qe on the map during the growth phase.

$$d = argmax_i(\parallel m_e - m_i \parallel), m_i \in N_e \tag{4.5}$$

The dissimilar neighbor d is the cell that is the most different of all the neighbors of the error cell, N_e.

The new row or column is filled in with model vectors that are the average of the model vectors on either side.

4.2.2 Map Stopping Criterion

The decision of when to stop training the map also involves the quantization error. The stopping criterion for a map is a function of its mean quantization error (MQE).

$$MQE_m = \frac{1}{n_U} \sum_{i \in U} qe_i, n_U \in \mid U \mid \tag{4.6}$$

The MQE of a map is the mean of all cells' quantization errors for the subset U of the maps' cells onto which data is mapped.

The stopping criteria that is used to decide when a map fits its data well is:

$$MQE_m < \tau_1 \cdot qe_u \tag{4.7}$$

where qe_u is the qe of the cell u in the upper layer. For the first-layer map, u is treated as a single cell through which all input into the first-layer map flows, and its quantization error, qe_0 keeps track of how diverse the data the map has received is.

The parameter τ_1 serves as a control on the granularity of the representation of the data.

4.3 Clustering

Clustering is a general term for a large class of unsupervised learning techniques. The goal of clustering is to discover groups of data points or "clusters" that are similar to each other. A cluster is a set of one or more objects that we are willing to call similar to each other.

The Unweighted Pair Group Method with Arithmetic Mean (UPGMA) style of clustering is one of the most popular and simple methods used (Romesburg 1984). Its output is a tree, as shown in figure 4-3. Each branching point in the tree corresponds to a cluster. The closer to the top of the figure, the fewer clusters you have, and the more data points each cluster includes. The full tree for a data set is sometimes thought of as its "similarity structure".

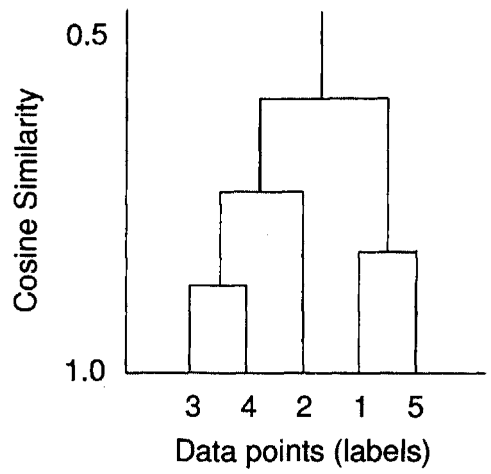

Figure 4-3: A cluster tree produced by UPGMA. Data points 1-5 are clustered. Clusters are indicated by crossbar junctions. The height of the crossbar corresponds to the similarity level of the cluster. The clusters in this diagram are: (3 4), (1 5), (3 4 2), and (3 4 2 1 5)

4.3.1 The Basic Algorithm

1. Compute resemblance matrix.

   Given a table of data points, a resemblance matrix stores the similarities between every pair of points. The rows and columns of the matrix are labeled with the identifier of the data point, typically an index. Due to symmetry, the matrix only stores values in the lower diagonal, and keeps null values everywhere else. The similarities can be computed with a measure such as Euclidean distance, for which more similarity means a smaller number, or cosine similarity, for which more similarity means closer to 1.

2. Find the most similar pair of points.

   The cell in the matrix with the most similarity indicates the most similar pair of points. The labels of the corresponding row and column indicate the pair of points.

3. Create a new cluster from that pair.

   This pair of points become a new cluster. The similarity value of the cluster is noted, and the cluster label is the combination of the labels of the old points. The cluster is treated as the midpoint between the previous points.

4. Merge the pair into a single row and column in the resemblance matrix.

   The two rows and two columns that corresponded the most similar pair are deleted from the resemblance matrix. A new row and column are added after the remaining ones. It is labeled with the combination of the labels of the old points.

5. Recalculate the similarities for the new row and column.

   All the old values in the matrix are left unchanged. The values in the new row and column represent the similarity between the new cluster and each remaining point. For each remaining point, the new value will be the average of two values from the discarded rows and columns. If the old matrix stored information about points 1, 2, 3, 4, and 5, and the new matrix stores information about points 1, 2, 5, and the new (3 4) point, then the similarities will be found in the following way:

$$similarity_{1(34)} = \frac{1}{2}(similarity_{13} + similarity_{14}) \tag{4.8}$$

$$similarity_{2(34)} = \frac{1}{2}(similarity_{23} + similarity_{24}) \tag{4.9}$$

$$similarity_{5(34)} = \frac{1}{2}(similarity_{35} + similarity_{45}) \tag{4.10}$$

6. Repeat steps 2-5 until there are is only one row and column left in the resemblance matrix.

Three strategies can be employed to cut the tree at the appropriate level of similarity to produce a good set of clusters. First, the number of desired clusters can be specified. Secondly, a fixed percentage of the range between the similarity of the most similar cluster and the least similar cluster can be used. Thirdly, the tree can be cut at some point where the size of the similarity range between clusters is the greatest. For example, in figure 4-3, the range between the (3 4 2) cluster and the (3 4 2 1 5) cluster is the greatest because the is the most distance between their crossbars. The idea behind this is that the larger the distance, the more separated the clusters are in their space. These three strategies can be mixed and matched to produce and optimal set of clusters.

4.4 Cluster Association

The algorithm used to associate clusters together is extremely simple. It is based on a simplified notion of hebbian learning, the idea that the synaptic strength of the connection between two neurons increases the more often those two neurons fire at the same time.

Each cluster in a map has a unique cluster ID. For all maps that are associated together, any clusters that are active at the same time get a counter incremented on their combination.

For example: map A has clusters A_1, A_2, and A_3. Map B has clusters B_1, B_2, and B_3. Let us say that at a given step, A_2 and B_1 are both active as the result of input coming into those maps which are part of those clusters. A link is stored between A_2 and B_1 and a counter on that link keeps track of how many steps there are in which those two clusters are both active.

The most frequently co-active cluster from map B for a given cluster A_i is considered to be the cluster to which A_i is associated with. The associated cluster to A_i in B may change over time if a new cluster begins to co-occur with A_i more than the old one had.

Chapter 5

Architecture and Implementation

This chapter discusses the architecture and the implementation of a computer program demonstrating the model of intrinsic representation.

5.1 Architecture

The architecture of the program is structured into four major areas:

1. A Blocks World Simulation

2. A Self Organizing Map

3. A Cluster Set

4. Cluster Associations

Each of these blocks feeds into the other in succession. Below, connections between these areas as well as the areas themselves are discussed.

5.1.1 Blocks World

A blocks world is, at its most basic, a problem domain. Typically it includes blocks of various colors and shapes, capable of being placed in different locations in a space.

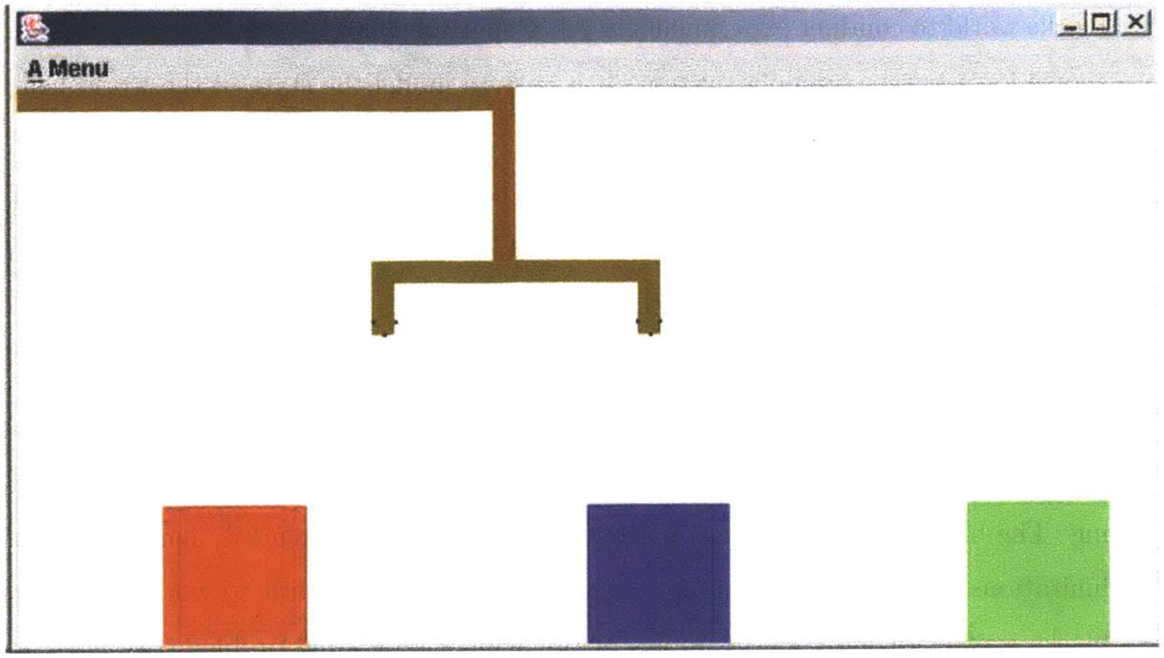

Figure 5-1: A simple 2D Blocks World.

Blocks are capable of being stacked upon one another if they are properly shaped. Typically a robot arm is used to manipulate the blocks.

The most well-known implementation of an AI system using a blocks world as a problem domain was SHRDLU, a program developed by Terry Winograd at MIT (Winograd 1971). The program's focus was on language interaction with a system that could do limited reasoning about a block's world. The system was able to solve a problem set related to the blocks world. It could answer questions and carry out commands.

While Winograd's world did prevent cubes from being stacked on top of pyramids, it did not attempt to model the physical reality of the problem set much more faithfully. The arm could pick up any block that did not have something else resting on it, without concern for grip. The arm never placed a block in a position where it could be affected by gravity, thus gravity was not simulated. The arm was essentially prevented from making any "mistakes" by the reasoner, and this simplified the model greatly.

Since then, various symbolic AI systems have used problem spaces similar to a blocks world to conduct experiments in AI (Copeland 2000). A blocks world is a good choice for a purely symbolic system. It is easy to model the state of the world using well-defined symbols.

More recently, the notion of a blocks world is being taken more literally, and experiments are being conducted using blocks worlds that more faithfully model Newtonian Dynamics (Beule, Looveren, and Zuidema 2002).

Why Choose the Blocks World? There are several reasons that the blocks world makes sense to use as the problem domain for a project concerned with symbol grounding. The blocks world as it was originally used is, pun not intended, symbolic of the limitations of symbol-only systems. Traditional AI systems built to work within the blocks world provide good examples of the symbol grounding problem. While they were successful within limited domains, systems like SHRDLU could not scale up to incorporate more complex worlds.

Once Newtonian Dynamics are added to the blocks world, the domain provides a reasonably good analogue to those that infants are presented with. A system interacting with objects in this domain can be thought of as an infant at a particular level of development. This allows using faithfulness to an infant's true behavior as a criterion for success. Unlike game domains such as chess or expert domains such as medical diagnosis, the blocks world allows the comparison of brains that are not at advanced levels of development. Operating from the bias that we need to understand how simple skills are formed before we can understand how complicated skills work, a domain that simple minds can understand is desirable.

When you consider what it would take to make a robot that plays with blocks, you begin to realize that even this seemingly simple domain has enough variability to be an interesting problem space. (Minsky 1988) used the idea of a blocks world extensively. In his theories, a child at play with blocks learns problem solving skills that are used to solve different problems later in life. In order to build towers out of blocks, children need intuitive understandings of simple physics, how to make wholes from parts, etc. Thus, there is hope that we may learn how to make systems reason about more complicated domains by having them start reasoning about less complicated domains.

5.1.2 Blocks World to Self-Organizing Map

Data flows into the system from the environment. A simple 2D blocks world provides the environment for the present system. In the blocks world, there is an arm with a simple grip useful for grasping at objects, much like the arcade novelty that allows players to try to catch prizes by controlling a robot arm. There is also an eye, whose focus is represented by a square that moves around the environment. The eye and the arm are the sources of sensory and motor interaction with the world. There are also blocks in the world able to be picked up and stacked on top of one another. Simple physics modeling is in place to enable such constraints as gravity.

Both the eye and the arm are equipped with sensors. Every sensor is normalized, and thus reads a real value between zero and one. The eye has retina sensors arranged in a 2D array that register the red-green-blue value of any spot they are over. The eye also has sensors that report its horizontal and vertical orientation, mimicking the information the brain receives from the muscles that move the eyes.

The arm has proprioceptive sensors and tactile sensors on its grip. The proprioceptive sensors report how far it is extended both horizontally and vertically, mimicking feedback from muscles and joints. The tactile sensors report if an object is colliding with the sensor or not.

A vector of sensor values is read out from the eye and arm each time their values change. The vector has n dimensions, where n is the number of independent sensors on the device. As mentioned, this vector is normalized. Thus, its relevant information can be thought of geometrically as its angle in an n-dimensional space.

As vectors are read out from a given device, they are received by a self-organizing map. For this project, two self-organizing maps are used; one for the eye and one for the arm. The map is specialized to its input only in the respect that its dimensionality must match. As vectors are received, the map's self-organizing process iterates in the way described in section 4.1.1.

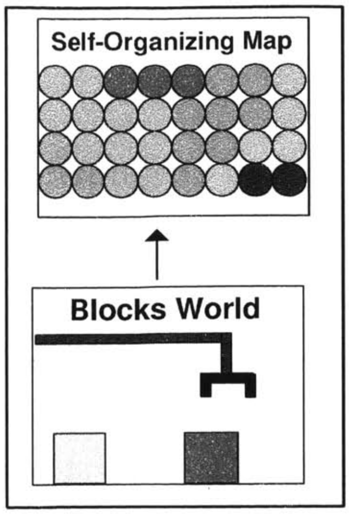

Figure 5-2: Blocks World to Self-Organizing Maps. Data from the blocks world is read out into a self-organizing map for each modality in the form of real-valued normalized vectors.

5.1.3 Self-Organizing Map to Cluster Set

The maps are allowed to self-organize for a significant amount of time. Once the map stop criterion is reached (equation 4.7), the map is switched to read only mode. Using the model vectors in the cells of the 2D array, a clustering algorithm is used to separate the major clusters in the space. This is necessary because while the map arranges data in continuous regions, it does not group data into discrete sets.

Each of the clusters on the map is given its own unique identifier, and a new data object is created to represent it. This data object keeps track of the similarity measure of the cells inside the cluster, as well as the indices of the cells it contains. Clusters are stored together in cluster sets, which also keep track of the relationships between clusters.

5.1.4 Cluster Set to Cluster Association

Once clusters have been stored for a map, they are associated with clusters in other maps. Clusters are never associated with clusters from the same map. The association process is extremely simple. When clusters are active at the same time, a counter

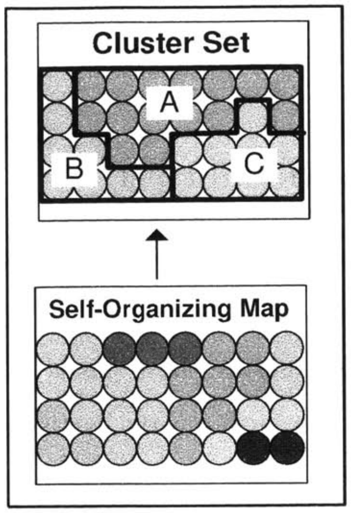

Figure 5-3: Self-Organizing Map to Cluster Set. A map is divided up into clusters A, B, and C using a clustering algorithm that operates on its cells.

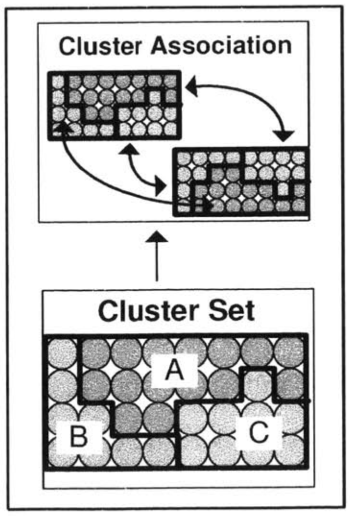

Figure 5-4: Cluster Set to Cluster Association. Clusters in different maps are associated together.

labeled with the names of those clusters is incremented. Clusters are active when input matches most closely with a cell within the cluster. The cluster association system identifies the clusters most associated with any particular cluster.

5.1.5 Modality A to Modality B

Once the system has been trained, it can be used to send signals between modalities and exhibit behavior that neither modality separately would be able to accomplish as easily by itself. Figure 5-5 illustrates how this is possible for the simple example of getting the arm to move to the position the eye is currently looking at.

5.2 Three Levels of Representations

The architecture I have just described takes advantage of three different levels of representation. While they have been mentioned before, it is worth re-emphasizing them. The following describes these representations in greater detail.

Map Cells Each cell keeps track of a subset of information that enters from the environment during the SOM process (section 4.1.1). Each cell in the map stores an n-dimensional vector. The data contained in that vector corresponds to a point in an n-dimensional information space.

In the context of a self-organizing map, a map cell represents a point in an n-dimensional information space that is representative of a class of points that the map has observed. Because a trained self-organizing map presents a lateral organization of the regularity in the data presented to it, each cell will roughly correspond to a point of statistical interest rather than a random sample of the space.

Clusters The cluster is one representational level higher than a map cell. Clusters are discovered in a trained SOM through the process described in section 4.3. Clusters are collections of statistically similar cells in a SOM. Because cells represent vectors in

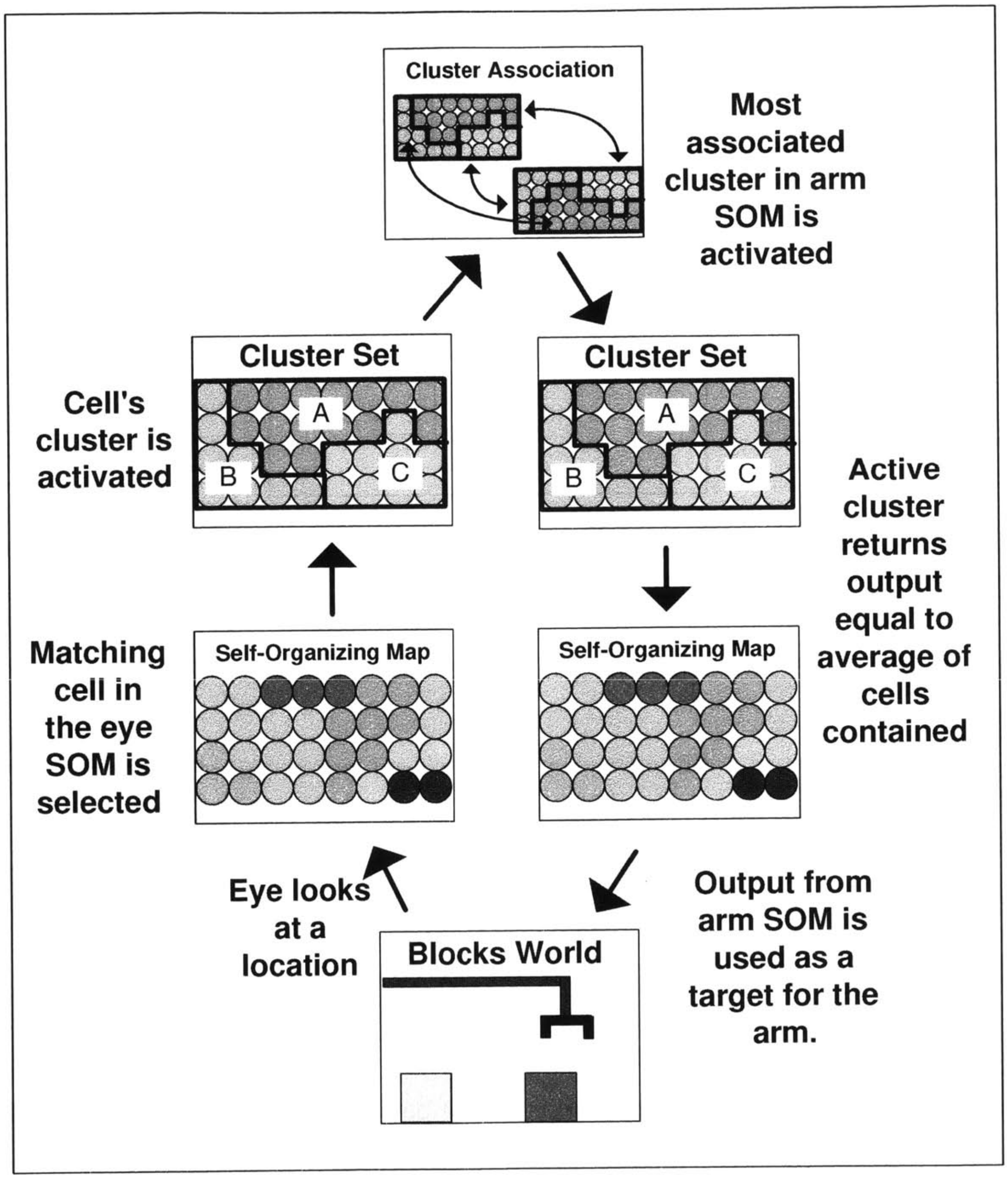
Cluster Association
Most associated cluster in arm SOM is activated
Cluster Set
A
B
C
Cell's cluster is activated
Cluster Set
A
B
C
Active cluster returns output equal to average of cells contained
Matching cell in the eye SOM is selected
Self-Organizing Map
Self-Organizing Map
Eye looks at a location
Blocks World
Output from arm SOM is used as a target for the arm.

Figure 5-5: An example of a trained representation. Representation has been trained with the eye fixated on the arm and no blocks. Arm and eye were moved together throughout the space. Clusters were formed, and further arm-eye movement was used to create associations between the clusters. This shows how the arm can then move to a location the eye is looking at by using the trained representation.

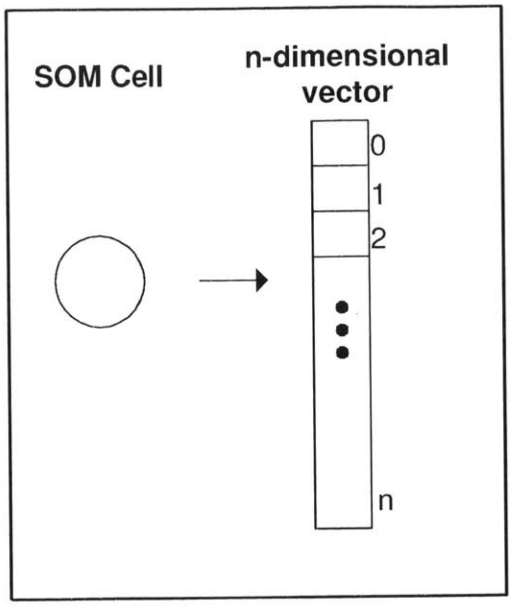

Figure 5-6: Each SOM cell stores an n-dimensional vector that represents a point in an n-dimensional information space.

an n-dimensional information space, a cluster therefore is a collection of statistically similar vectors.

In the context of a self-organizing map, a cluster represents a collection of cells that have been arranged near each other by the SOM process. Remember that the SOM process also arranges cells such that dissimilar cells are distant in the space. As a result, the cells inside a cluster are similar to each other, but different from other major clusters in the map.

Putting this together with what we know about a map cell, a cluster represents a "subspace" of an n-dimensional information space by storing a collection of n-dimensional vectors. While these vectors are not a basis for that subspace in a linear algebra sense, they are a collection of statistically interesting vectors that are representative of a class of points in that subspace.

Associations/Symbols Associations are a representational level higher than clusters. An association, as depicted in figure 5-4 by arrows, represents a collection of clusters, usually two. Associations are discovered between different maps according

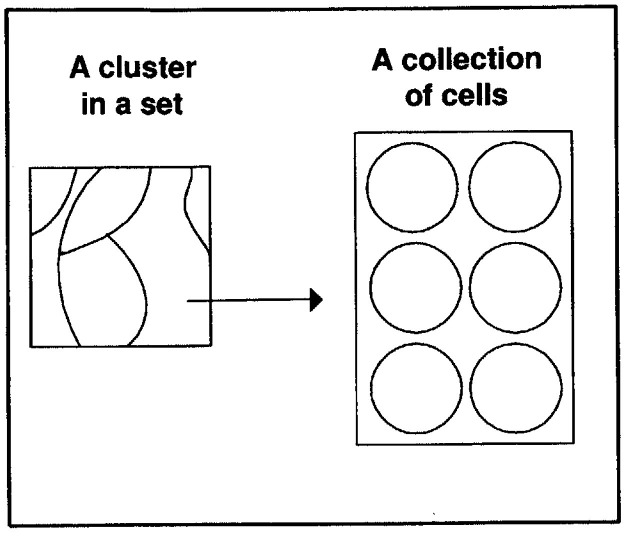

Figure 5-7: A cluster in a SOM represents a collection of cells, which in turn represents a collection of points in an n-dimensional information space

to the process described in section 5.1.4. This creates associations between clusters that are active at the same time in different maps.

Associations by themselves provide a bridge between clusters in different information spaces and capture relationships between different sensory modalities. Networks of clusters are formed by using associations as the links.

Associations can ground symbols. Let us take, for example, a linguistic symbol, `(Red_Block)` , and let us say that the symbol can be either active or in active. Associations are created between `(Red_Block)` and the appropriate clusters by a supervised learning process that activates the symbol when the sensory systems are focused on the red block. Once created, they can be used to key into information on the sensory maps. In the other direction, incoming sensory information that falls within a cluster can be used to summon the appropriate association.

5.3 Learning About Blocks

In this section I demonstrate how my system learns about blocks in a simple 2D blocks world at all three of the representational levels just described. In a nutshell, the system discovers implicit representations of blocks in its world in an unsupervised manner. Then it assigns explicit symbols to those implicit representations in a supervised manner.

The first experiment involves two self-organizing maps that read data from a blocks world. One map reads data from the eye in the blocks world, and the other map reads data from the arm. Figure 5-8 illustrates the features that make up the input data vectors.

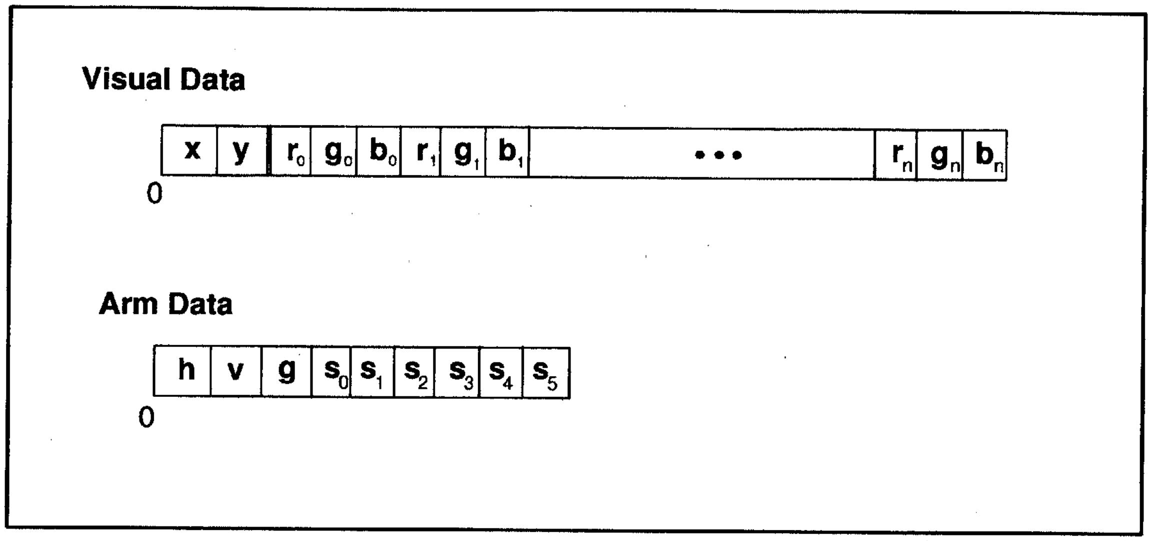
Visual Data
x
y
r_0 g_0 b_0 r_1 g_1 b_1
•••
r_n g_n b_n
0
Arm Data
h
v
g
s_0 s_1 s_2 s_3 s_4 s_5
0

Figure 5-8: A guide to the features stored in the visual and arm data vectors that arrive at the eye map and the arm map respectively. The visual data includes x and y coordinates for the eye position, followed by r, g, and b which store the red, green and blue values for each pixel position in the eye. The arm data includes h, v, and g which store the horizontal and vertical components, the width of the grip, and 6 sensor values that are read from sensors on the gripper/hand. All values are normalized between 0.0 and 1.0.

Figure 5-9 shows the state of the system at start-up of the blocks learning process. The blocks world view shows a robot arm, three differently colored blocks, and a rectangle that represents the current focus of the eye. There are two maps visible above the blocks world view to the right and the left. The one on the left is a visual representation of the eye. A zoomed view is shown in figure 5-10. The purpose of the graphics in each cell of the map is to visualize the state of the underlying model vector. The graphic for the eye map displays a bitmap as well as two numbers for the x and y positional data.

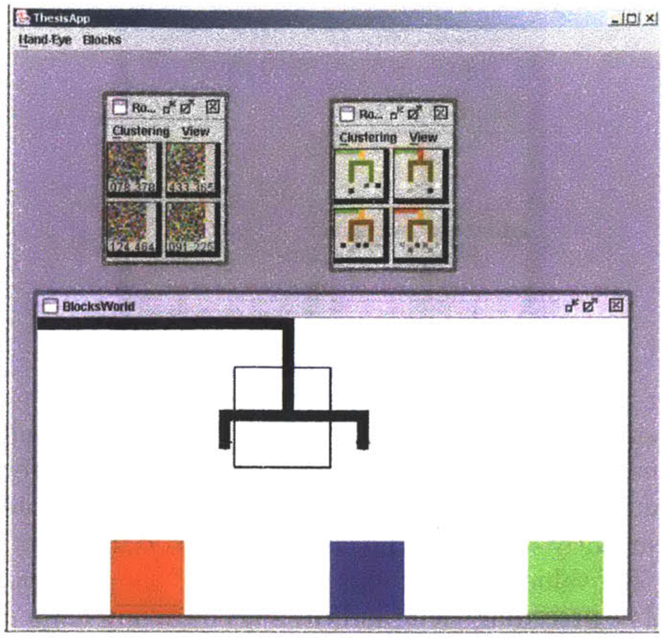

Figure 5-9: My system upon start up. A simple 2D Blocks World and two GSOMs are visible. No training has occurred.

The map on the right is a visual representation of the arm. A zoomed view is shown in figure 5-11. The graphic for the arm map displays a cartoon of an arm that is shaded to represent the extension of its "limbs". The arm map also displays six boxes that represent the values of the sensors that are on the arm gripper/hand.

Both maps start out as a 2-by-2 array of cells. As the maps train, they grow in size according to the GSOM algorithm (section 4.2), which grows the map in the areas where its representation of the incoming data is the most dense.

5.3.1 Training The Eye Map

First, the eye map is trained on the objects in the scene separately. To accomplish this, the eye is shifted between the objects in the scene several times. While viewing a single object, the eye is allowed to scan the object by taking a circular path around it. As the eye moves from object to object, the eye map grows to capture more of the views it is receiving. Figure 5-12 illustrates the progress of the eye map in this phase of training.

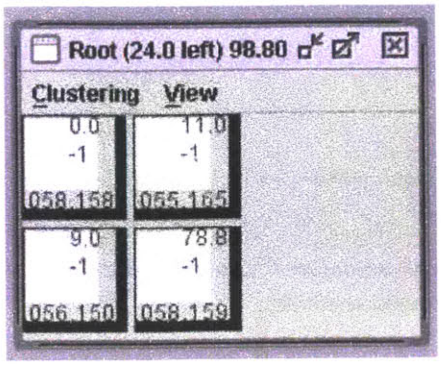

Figure 5-10: A visual representation of the eye map trained on white space just after initialization. Currently the eye map is only 2 by 2 but will grow as the eye map trains. The top number in each cell is its quantization error, the middle number is the id of its cluster (currently -1 because no clustering has occurred yet), and the bottom numbers read out the values of the x and y features of the cells model vector. In the title bar, the number preceding "left" indicates l, how many units until the map stop criterion and the number to the right indicates the mean quantization error for the entire map.

The result of this process is a map that captures views of the objects in the scene.

5.3.2 Training The Arm Map

Once the eye map has been trained with the stationary objects, the arm is trained on the blocks in the scene by being moved to each block in succession, grasping it, picking it up, moving it around, lifting it up, and eventually dropping it[1].

The result of this process is shown in figure 5-13 and illustrates a distinction between the arm with something in its grasp, and the arm without something in its grasp.

5.3.3 Eye Map With Interactions

While the arm is being trained, the eye is focused on the actions of the arm, thus updating the eye map to include the interactions between the arm and the blocks. Figure 5-14 illustrates the resulting map.

[1]In the implementation completed by submission time, the arm picked up and moved the blocks extensively in the y direction, but only slightly in x. I see every reason to believe that if the arm map can generalize across the y direction, it can also generalize across the x coordinate.

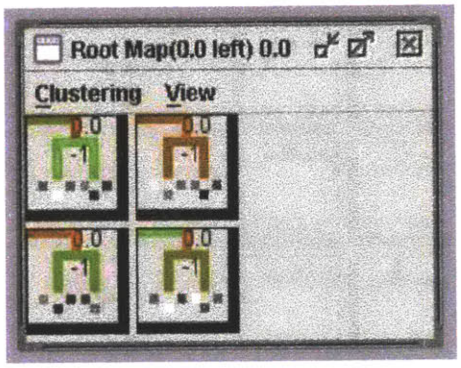

Figure 5-11: A visual representation of the arm map just after initialization. Currently the eye map is only 2 by 2 but will grow as the eye map trains. The top number in each cell is its quantization error, and the lower number is the id of its cluster (currently -1 because no clustering has occurred yet). Within each cell is a cartoon representing the arm. Each segment of the cartoon arm is colored from red to green based on the values of the underlying model vector. For example, a model vector representing the arm in the bottom right corner of the map would have a light green horizontal component and a light green vertical component. A model vector representing the arm in the upper left corner would have both components bright red. The gripper/hand part of the cartoon also changes colors related to its representations, ranging from green to red as it constricts. Underneath the gripper/hand are six small rectangles that indicate the binary values of the sensors on the arm, white for off, black for on. In the title bar, the number preceding "left" indicates l, how many units until the map stop criterion and the number to the right indicates the mean quantization error for the entire map.

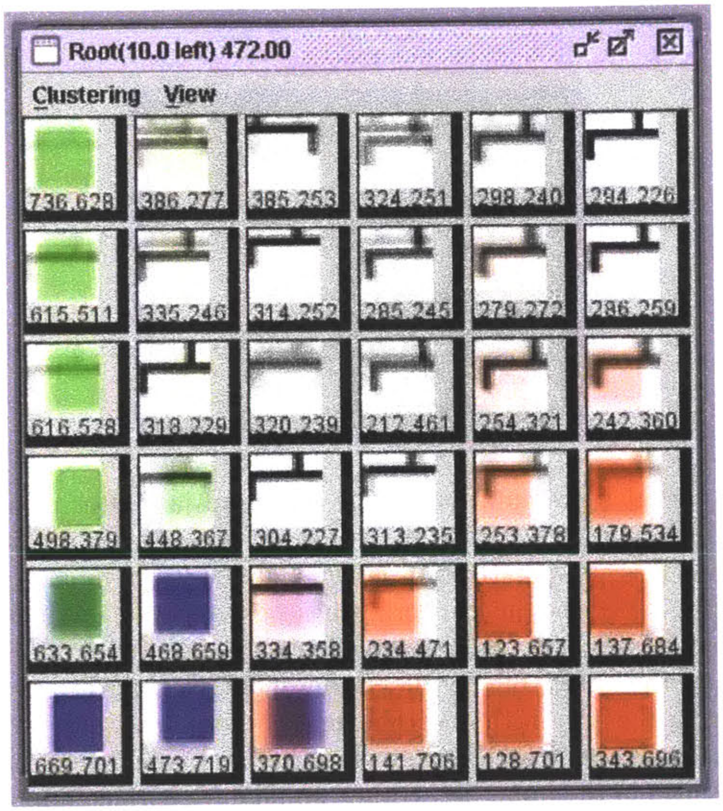

Figure 5-12: A visual representation of the eye map partially trained on the four objects in the simple 2D blocks world. The map has now grown to better fit the data. In the bottom left, the map shows a few views of the blue block. In the bottom right several cells show views of the red block. In the upper right, middle and center, several views of the gripper/hand can be seen. The cells along the upper and middle left show views of the green block. The "smeared" cells are the product of different inputs that came soon after one another and thus created cells that are on a cluster border. The title bar shows that there are 10 units left until the map stop criterion.

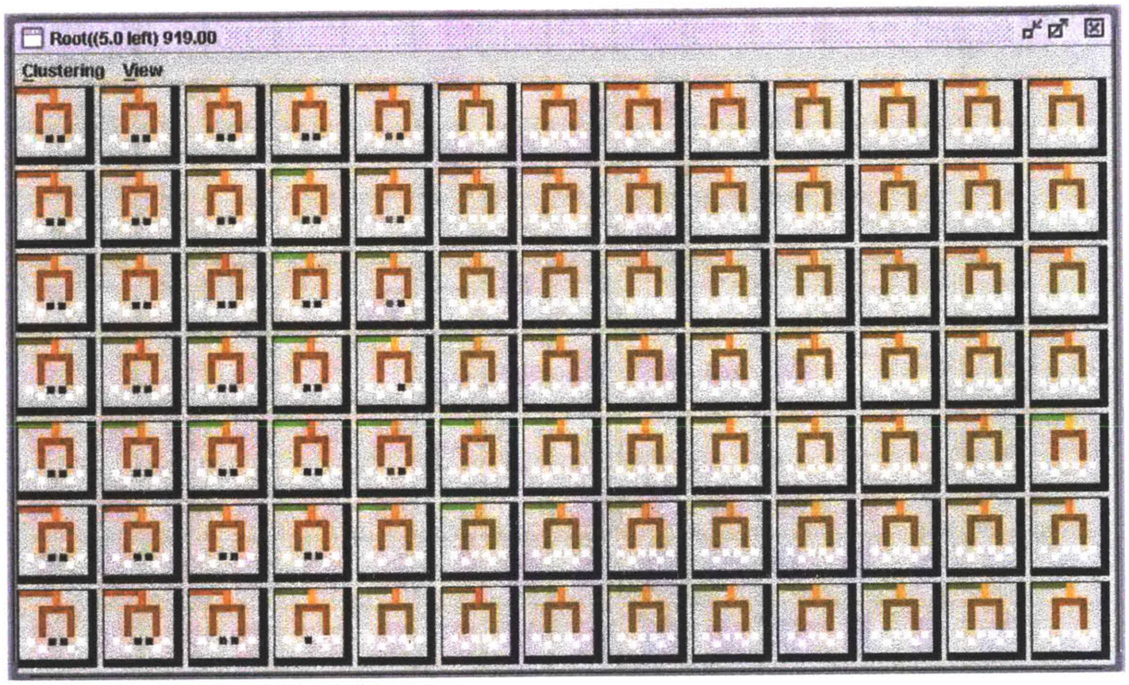

Figure 5-13: A trained arm map. A clear distinction between cells representing the hand while the middle sensors are active (when it is holding a block) and while they are not is seen from left to right. Within these two groups, there is a relatively smooth distribution of arm location.

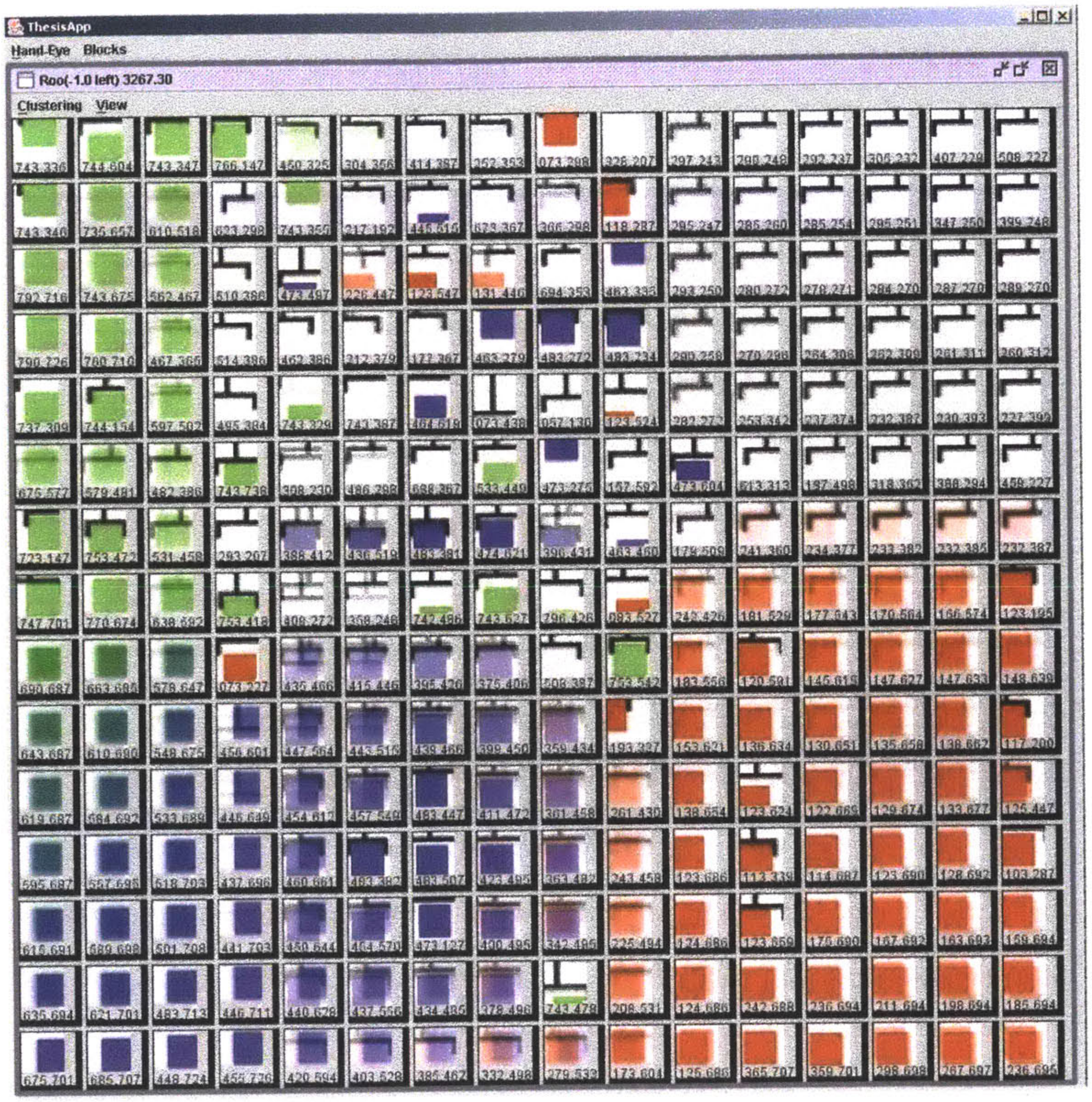

Figure 5-14: A trained eye map, complete with views of the arm interacting with blocks.

Training of both maps continues until the map stopping criterion is met (equation 4.7). The map stopping criterion provides an objective standard for when to end the training process because it signifies that the map has met an adequate fit to its data. While the hierarchical version of the algorithm calls for children maps to be spawned off at this point, my implementation ends the process here to simplify the visualization.

My implementation reduces the size of the neighborhood, $N_c(t)$ (section 4.1.1), while the system trains. The following equation defines l, the value used to compute the neighborhood function.

$$l = MQE_m - \tau_1 \cdot qe_u \tag{5.1}$$

5.3.4 Clusters

Once the eye map and the arm map have been sufficiently trained, a clustering algorithm is run on their model vectors (see section 4.3). The algorithm creates groups of cells that are similar to each other, and less similar to other cells on the map. Figure 5-15 shows the eye map segmented into clusters, and figure 5-16 shows the arm map segmented into clusters.

5.3.5 Associations

Once the maps have been segmented into clusters, their clusters are associated together. The training process is a repetition of the arm training—the arm travels to each block, grips it, picks it up, and moves it around, while the eye looks on.

During *this* training, both maps are given the opportunity to form associations with a special utterance map, which contains symbols that are activated or deactivated during training. When the eye and the arm are interacting with a block, the appropriate symbol is activated. When the eye and the arm leave the area of the blocks, no symbols are asserted. The symbols are `(Red_Block)`, `(Blue_Block)`, and

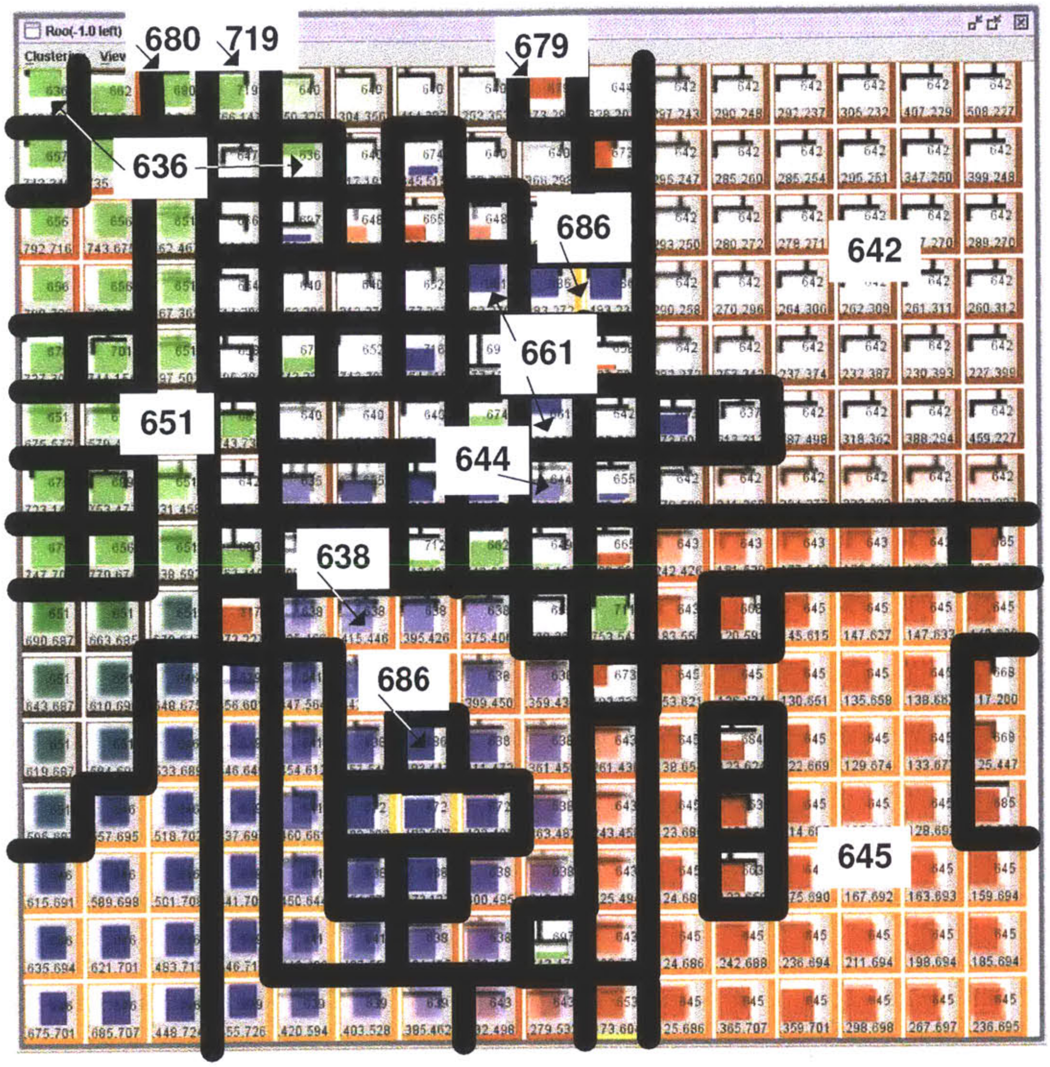

Figure 5-15: A trained eye map with clusters overlaid. The cluster IDs of relevant clusters are highlighted.

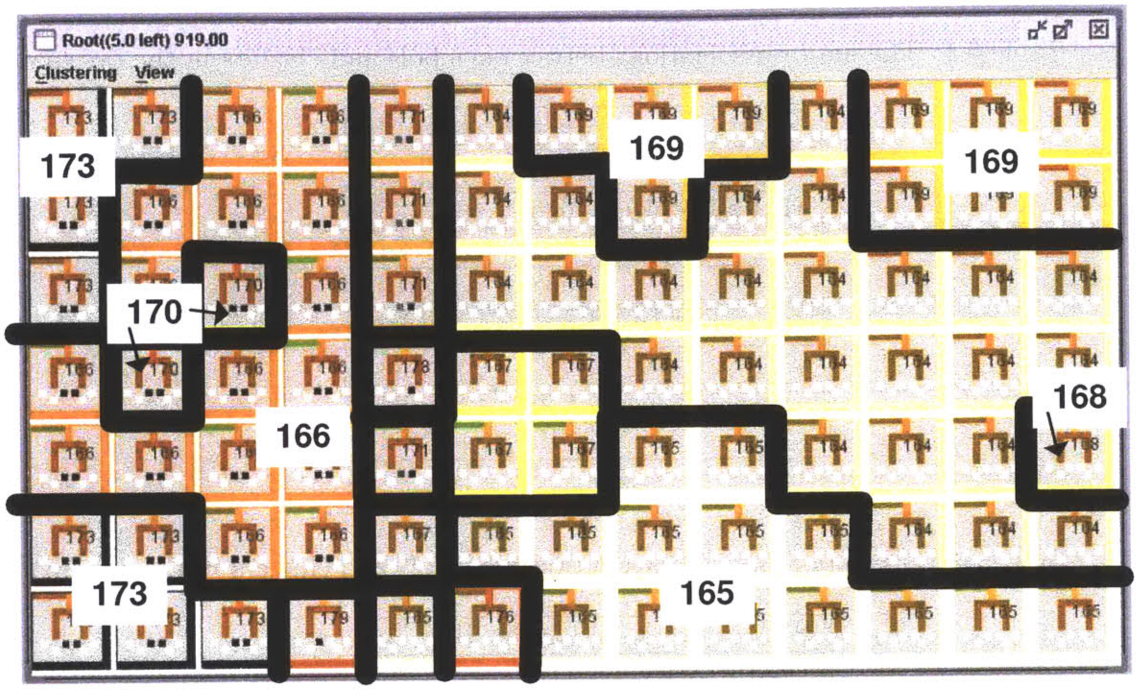

Figure 5-16: A trained arm map with clusters overlaid. The cluster IDs of relevant clusters are highlighted.

(Green_Block). These symbols are undivided; while to us they imply a combination between a color and a shape to a human reader, to the system, they are only treated as distinguishable atomic tags. This is consistent the symbol grounding problem, whose elaboration helps us to realize that symbols by themselves are just tags until associated with sub-symbolic meaning.

Table 5.1 shows the results of the association between these symbols and the clusters in the eye map. Higher frequency corresponds to greater association between the symbol and the cluster. The cluster IDs are the same as the cluster IDs in figure 5-15 to provide the reader a sense of the clusters that the symbols are being associated with. Table 5.2 shows corresponding results for association between the symbols and the clusters in the arm map. These cluster IDs correspond to the ones in figure 5-16.

By design, both the eye and the arm clusters are associated with the same set of symbols. This is because the kind of learning taking place is intended to simulate a child's association of utterances with sensory information. This models the theory

Table 5.1: The major associations between symbols and clusters in the eye map

SYMBOL	Cluster ID	Frequency
(Red_Block)	645	69
(Blue_Block)	686	35
	661	24
	638	10
(Green_Block)	651	80
	680	26
	636	23
	719	20

Table 5.2: The major associations between symbols and clusters in the arm map

SYMBOL	Cluster ID	Frequency
(Red_Block)	173	78
	169	35
(Blue_Block)	166	63
	165	22
	170	14
(Green_Block)	166	100
	168	10

that a child can make associations between its visual input and its linguistic input and between its arm's input and its linguistic input independently.

Generally, the associations tend to fall into the largest clusters that, to the eye, appear to be the most representative of the block in the symbol. In this case, however, table 5.1 shows that the symbol (Blue_Block) is instead most associated with a smaller cluster, though the largest cluster corresponding to the symbol (Blue_Block) appears farther down on its list. This demonstrates how even small clusters can represent important parts of the information space.

One of the interesting results of table 5.2 is that the (Blue_Block) symbol and the (Green_Block) symbol are both most highly associated with cluster 166. This cluster corresponds to a state where the arm is gripping something. The associations further down the list, which match to different clusters, dissociate the two symbols so that they can be independently identified.

This experiment demonstrates the ability of the system to ground symbols in sensory data. The system has acquired sensory data and organized them into classes in an unsupervised fashion. Later, this data was associated with symbols in a supervised fashion.

5.4 Associating Hand with Eye

In this section I demonstrate how my system uses associations formed between the position of the eye and the position of the arm to enable the arm to move to the current position of the eye.

5.4.1 Training

The training of the eye map and the arm map follows a simpler process in this experiment than in the previous blocks-learning situation. In this experiment, training proceeds by fixing the position of the eye to that of the gripper/hand, and moving them together to random points in the space. The space does not have any blocks into it.

What we expect from this training is for the eye map to have stored very similar images since it views the same part of the gripper for all time. Because of this, the x and y components of the eye data should become more significant. The resulting eye map should have an even distribution of x and y positions across it. The arm map will emphasize the horizontal and vertical components as they are the only ones that are varied.

5.4.2 Clusters

Figure 5-17 shows the trained eye map after the clustering process has run. Notable clusters are highlighted. Our expectation of a smooth and continuous map of x and y values has been met.

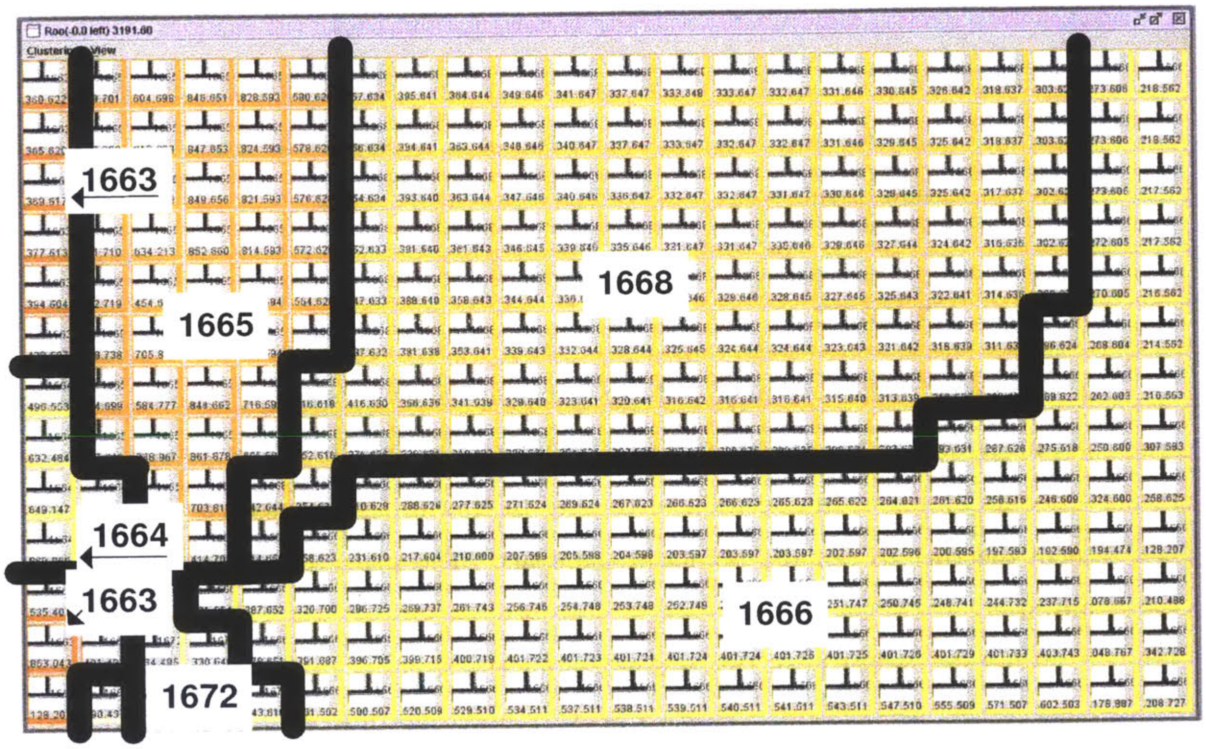

Figure 5-17: A trained eye map with clusters overlaid. Relevant clusters are highlighted. All the cells show closely similar views of the world, but the x and y components vary across the map.

Figure 5-18 shows the trained arm map after the clustering process has run, with notable clusters highlighted.

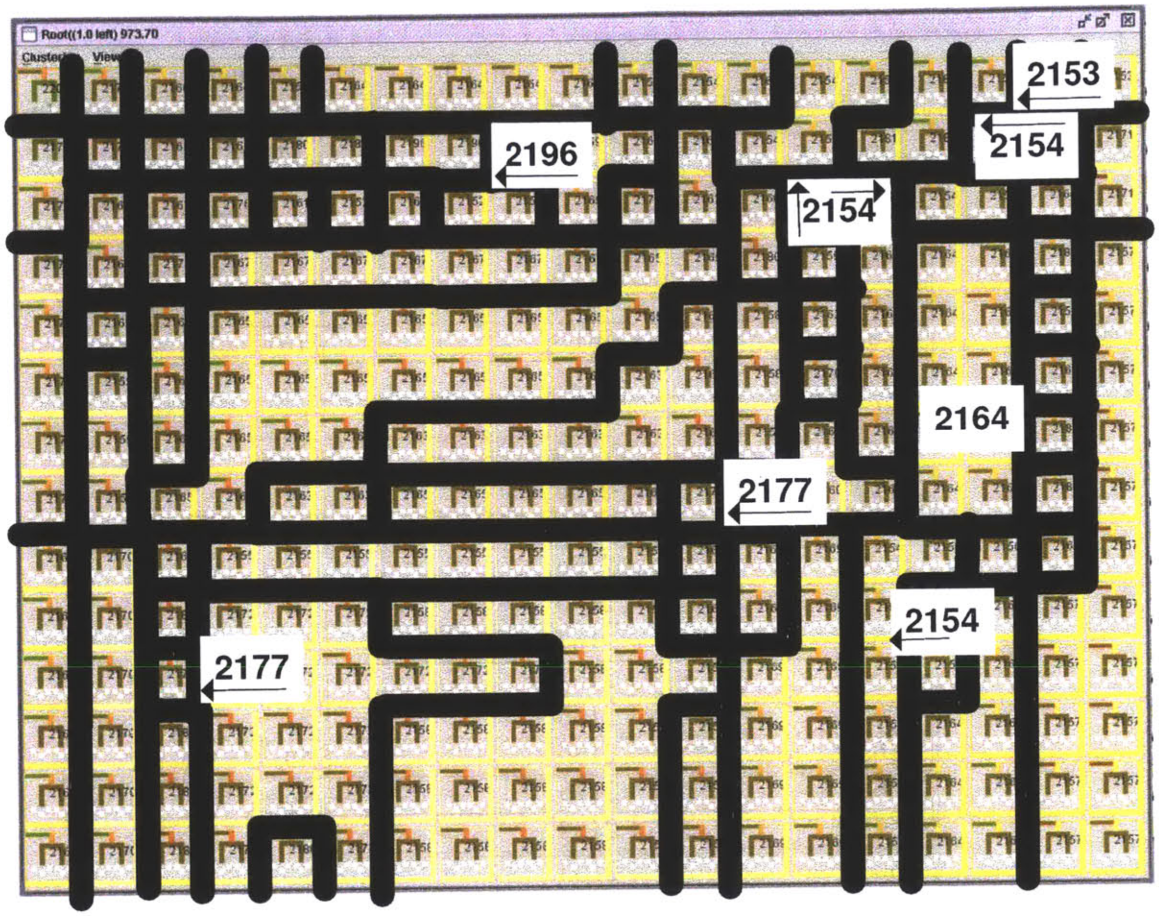

Figure 5-18: A trained arm map with clusters overlaid. Relevant clusters are highlighted. While the gripper sensors and grasp remain constant, the map self-organizes using the horizontal and vertical components of the data alone.

5.4.3 Associations

In this experiment, associations can form between the eye map and the arm map. In this case, clusters in one map serve as "symbols" for the other map, and vice versa. Table 5.3 shows the data from the cluster associations.

Table 5.3: Associations between the eye map and the arm map.

Eye Map Cluster ID	Arm Map Cluster ID	Frequency
1663	2177	152
1663	2154	140
1663	2164	136
1663	2153	127
1663	2181	125
1663	2196	125
1663	2169	101
1663	2152	92
1663	2156	81
1663	2155	78
1663	2166	72
1663	2187	63
1663	2163	58
1663	2180	57
1663	2167	56
1663	2157	55
1663	2165	52
1663	2186	49
1663	2176	45
1663	2182	45
1664	2153	27
1664	2181	24
1664	2152	20
1664	2187	20
1665	2196	20
1665	2164	19
1665	2181	15
1665	2153	15
1666	2177	36
1666	2154	33
1666	2181	29
1666	2180	19
1672	2196	17

5.4.4 Activity

These associations allow the arm to move into a region near the eye's view. To accomplish this, the currently active cluster in the eye map is identified using the current state of the eye. Then, the arm cluster most highly associated with the currently active eye cluster is selected from the arm map. The vectors from the cells within this cluster are averaged together, and the arm is driven towards this average vector. A feedback process is used whereby the arm continues to move in the direction of the average vector so long as its current state vector does not match it. Note that while this experiment focused on moving the eye to the arm, the inverse could also be accomplished with the same trained maps by simply reversing the process just described.

This experiment demonstrates a deeper use of the model of intrinsic representation—to enable action. Through an unsupervised process, the different coordinate systems of the eye and the arm are trained to be compatible, and to allow one to "speak the same language" as the other.

This experiment also shows how exchanges can be made between sensory systems without involving any linguistic symbols whatsoever. This observation expands what we mean by a symbol. The experiment demonstrates the equivalence of 1) associations made with clusters that store linguistic symbols and 2) associations made with clusters that store other kinds of sensory data. Of course, in the brain, linguistic symbols *are* just another kind of sensory data, so in a way, this is just as it should be.

At the same time, this experiment demonstrates the usefulness of the ability for linguistic symbols to probe into the domains of other sensory information. Perhaps the problem of generating language does not have to be solved as independently from other systems as we sometimes think. Perhaps almost all of the information is already present in the other senses, and all that is needed is 1) the right associations between them and the linguistic areas in order to reveal that information, and 2) a way of sequencing that information that follows the rules of a grammar. Conversely, in the other direction, one can see how a single linguistic symbol could become associated with a mass of clusters scattered throughout different information spaces and trigger the recollection of stored sensory states.

Chapter 6

Discussion

Now that I have presented the architecture and implementation of the model of intrinsic representation, let us go back and examine what has been accomplished. In chapter 1, I outlined the key features of my model. I will now review these features in light of what we have seen about the model.

- Collects information from its environment quickly and flexibly in a way that can be used to solve problems.

  We have seen that with the use of self-organizing maps as an adaptive representational system, information from a given sensory modality can be stored quickly. That data is organized in a flexible manner because it is accessible in a content-addressable way. We have seen with the example of the arm tracking the position of the eye, that such information can help systems learn how to solve problems that would otherwise require explicit instructions.

- Discovers and organizes simple relationships in the information that it collects.

  Self-organizing maps help to illustrate and organize the simple relationships between vectors in an information space.

- Uses the results of its organizing process to bootstrap higher-level descriptions, such as symbols.

By creating discrete clusters from the data collected by a self-organizing map, higher-level processes can either associate a symbolic tag with statistically interesting classes of data, or they can associate other classes of data with each other.

- Grounds the meaning of the descriptions it discovers in a semantics defined entirely by the system itself, thus creating *intrinsic* representation.

  As I mentioned in section 4.1, the self-organizing maps in my system create an implicit representation of a collection of sensory inputs. The descriptions of those sensory inputs in relation to each other are not designed, they are discovered. Moreover, the procedures that operate on those descriptions, namely the clustering algorithm that operates on a trained map, also create descriptions that are not designed, they are discovered. Thus, the system describes its inputs in a language of *statistical similarity*, rather than requiring *us* to describe them in a language of *syntactic rules.*

6.1 Symbol Grounding and Intrinsic Representation

What progress does this model make towards solving the symbol grounding problem? Does the representational system bring us closer towards the criteria of semantic completeness and knowledge generation? I would argue yes on all counts.

As discussed at length in chapter 2, the symbol grounding problem makes us realize that symbols are generally disconnected from the kind of meaning that people impose on them. The model of intrinsic representation does not have this flaw. The meaning of symbols—the clusters formed from regularity in the environment—is directly connected to the experience of the system. In fact, the meaning of symbols cannot exist without experience, as they are created through a process of experiencing the world. This is because without a cluster to associate with, a symbol is meaningless. Rather than placing the symbolic tag at the center of the representation and allowing meaning to come later, intrinsic representation places the meaning first, and allows the tag to come later.

6.2 Semantic Completeness

As described in section 2.3.1, semantic completeness is the goal of creating representations that are both semantically broad and semantically deep. Recall that semantic breadth refers to the ability to capture information about a broad range of topics. Semantic depth refers to the ability to capture knowledge about a single topic at a fine resolution.

The model of intrinsic representation specifically addresses the concern of semantic completeness. The ability to integrate information seamlessly from multiple information spaces provides a potential for semantic breadth that is not possible with a localized representation. Any representation that tries to account for all kinds of information spaces with a single set of conventions is doomed to either be too narrow, or over-general. By assigning separate maps to different information spaces, however, a healthy amount of resolution in each space can be captured. This division of labor also enables greater semantic depth due to the ability for each map to create a detailed representation of the space it is aimed at.

6.3 Knowledge Generation

In section 2.3.2, I said that knowledge generation should also be a goal of representations that seek to overcome the symbol grounding problem. I defined knowledge generation as the ability for a representational system to give us back significantly more information than we put in.

The model of intrinsic representation addresses the concern of knowledge generation head on. By combining the implicit representation of the self-organizing map with the explicit representation of the cluster and by providing a systematic means of going from one to the other, I have enabled my system to generate more knowledge than is given to it. Specifically, the system starts out only knowing how to extract knowledge, and ends up adding to that the knowledge that it extracts. Once the symbolic tag has been associated with its meaning, the system does not have to be told the features or properties of a block in order to identify it. Once the eye and the arm have been trained to associate regions in their coordinate spaces, the system does not have to be told how to bring one to the other. Considering that the system is being trained unsupervised and by example only, it does appear as if it is taking away more than it is receiving from its designer.

Chapter 7

Contributions

In this thesis, I have:

- identified some limitations of modern representations that prevent them from being more general,
- explained the idea of an internal semantics,
- drawn a distinction between explicit and implicit representation,
- broadened the definition of representation to include both explicit and implicit varieties,
- described *semantic completeness*, a concept that enables comparison between representational schemes,
- built a model that addresses the concerns of symbol grounding and allows representations to be formed autonomously,
- elaborated the three key differences between the model of intrinsic representation and traditional symbol systems, which were:
  - Symbols' descriptions are discovered from the statistical processing of experience.

  - Symbols' descriptions are equivalent to statistical regularities found in information spaces.
  - Symbols' descriptions carry their context with them by being situated in information spaces.

- implemented a computer program for my model that serves as an existence proof that the model can be instantiated,
- demonstrated that the program is capable of learning symbols for blocks in a blocks world, and
- demonstrated that the program is capable of associating the movements of its eye with the movements of its arm.

Appendix A

Multidisciplinary Background

Because of my desire to see more unity in the brain sciences, I have included a review of three fields that are not frequently combined with AI research devoted to studying human-level intelligence. The following represents a snapshot of several important aspects of these fields.

A.1 Computational Neuroscience

Computational neuroscience (CN) takes the idea of representation as dynamics extremely seriously. Although there are several reasons to be interested in the field, that one is the most compelling as far as this thesis is concerned.

While artificial intelligence is a field derived from computer science and draws inspiration from neuroscience, computational neuroscience is the opposite. People in CN seek to understand the informational capabilities of biological neurons and neural networks. To accomplish this, they study output from real neurons using electrophysiology. They build mathematical and computational models of neurons and neural networks, taking inspiration from systems mapped out in the brains of animals.

Computational neuroscience is useful to artificial intelligence as a toolkit of components and design principles that can be used to build intelligent systems. This is true for several reasons. CN generates theories about brain systems that are on a scale smaller than those systems that AI theorizes about, creating an excellent opportunity for collaboration. One of the criteria for success in CN is biological plausibility. An AI system that uses CN components inherits this property, which adds credibility. Additionally, from an engineering perspective, maintaining a criterion of biological plausibility is useful because it provides design space constraint.

Another important perspective CN brings to AI is its embrace of systems that use neuron-plausible representations (i.e. vectors and matrices of real numbers to represent bundles of neurons). While such systems are likely not a "silver bullet" of artificial intelligence, they shed light on computations that neural systems may be capable of. This provides an important addition to an AI researcher interested in brain representations.

An important contribution CN is already making is its focus on computation as a dynamical system. In a dynamical system, you represent computations by a set of differential equations rather than a logical program. Such systems impose restrictions on the ways information can be processed. In particular, they treat representations and the methods that operate on those representations in a much more fluid way. Representations are usually maintained as constraints between conditions, rather than static objects, which introduces concerns such as stability and results in the creation of "fuzzier" systems. These issues make some kinds of computations more difficult than they are on a standard computation model. However, they make other kinds computations easier. Those algorithms that are convenient for a dynamical system to carry out should draw extra attention, and by so doing, can provide additional constraint into AI research.

The discovery of statistical regularity in large amounts of data is one class of algorithms that dynamical systems excel at. Feed-forward systems using error-driven learning find effective ways to represent complex input-output transformations. Principal Component Analysis, an important method for discovering the most important statistical features of a stream of data, is straightforward using dynamical systems (Hertz, Krogh, and Palmer 1991). One property of these systems that is subtle but important, is that their representations are frequently their own best description. After training, the hidden layer of a feed-forward neural network—a representation of some of the regularity in the input-output relationship of the network—is frequently difficult to categorize in terms of the input or the input. While it may be frustrating from an engineering perspective to be unable to fully interpret the system in a cell-by-cell reductionist basis, some feel that this demonstrates the importance of such models (Robinson 1992). This property causes us to focus on understanding these representations as part of a larger system of the constraints that caused them to come about. This shift away from descriptions in favor of the system that creates them may be an important way to analyze the activities of large populations of neurons at a higher level of abstraction. If we can replace a stream of data with a few rules that made them come about, it seems like we have better tools, not worse.

A.2 Systems Neuroscience

Functional magnetic resonance imaging and its cadre of related brain imaging techniques have provided a new dimension of insight into how brains function. One of the most basic findings from this research has been the observation that as you think about things, predictable parts of your brain become activated. Predictable, in this case, means that researchers have an ever increasing understanding of how stimulus relates to specific patterns of brain activation. Areas have been identified that contribute to the processing of categories of stimuli, such as faces, places, and tools.

What we see in these images are diverse clusters of brain areas activated as different mental processes occur. There is a growing understanding in the neuroscience community that the organization of these activations are best thought of as a network (Fuster 2003). Thus, depending on what your brain is doing at a given moment, different networks of brain regions will be activated, exchanging information, and causing activity.

We know more about the structure of the brain and the way it processes information than we sometimes think. While our understanding is still limited, the brain sciences are rapidly creating footholds that AI could use to generate its theories. For example, the following diagram illustrates a sketch of the broad areas of the cerebral cortex and their interconnectivity. The two most useful ideas here are the hierarchical organization of the cortex, and the separate hierarchies for sensory input and motor output that interact at different levels.

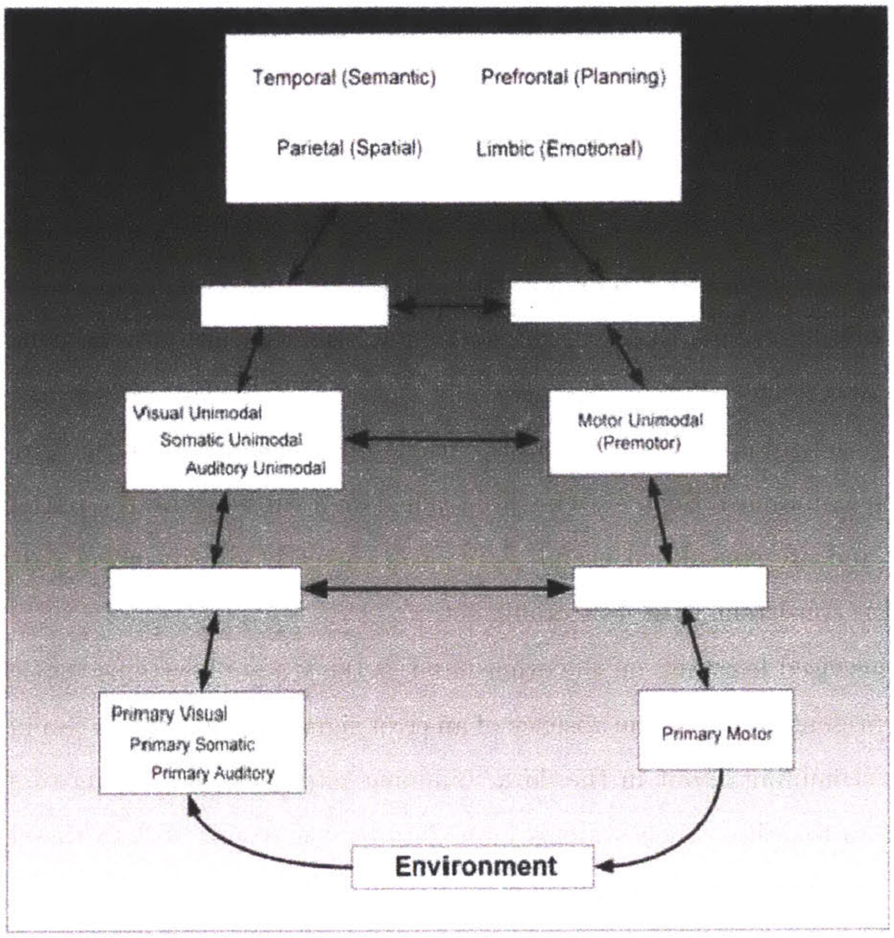

Figure A-1: The Perception-Action cycle. Modified from (Fuster 2003).

This hierarchical organization is not absolute. The brain has elements of heterarchy as well, allowing cross-cutting connections to be formed between layers. However seeing that an architecture of the cortex is beginning to emerge, it makes a lot of sense to allow this information to shape AI theories.

With these points in mind, it may make sense to think of a symbol as a "network of activation" that operates on top of this architecture. Such networks could be formed between or within any of the regions in figure A-1 and could have a large variety in their content and complexity. A single network could correspond to large assemblies of neurons and the connections between them. Much of the inspiration for such networks as representational units is present in neuroscience today (Fuster 2003).

A.3 Machine Learning

The field of Machine Learning, a member of the artificial intelligence family, concerns itself with two kinds of learning, supervised and unsupervised. Supervised learning is the study of systems that will categorize patterns that are presented to it in the presence of an error signal. One generally thinks about such systems as a student in the presence of a teacher who is responsible for showing it patterns and telling it what their names are. Such systems are trained on a subset of all the patterns it will ever see, and are considered useful if they can correctly classify novel patterns in a way that is consistent with its training set.

Unsupervised learning, on the other hand, is the study of systems that categorize patterns presented to it in the absence of an error signal. Such systems can be thought about as stumbling about in the dark, bumping into data, and trying to group the related data together. Such systems label data by the groups of data they can form, rather than by a teacher's instruction.

The field of data mining is a business application of unsupervised learning ideas. Its goal is to try and find correlations between large amounts of data that were previously obscure. For example, a data mining system might be set on a customer database for a large corporation, and might reveal a correlation between purchasing habits and season.

From a symbol grounding perspective, unsupervised learning is appealing. While supervised learning imposes a system of meaning by specifying an input-output relationship, unsupervised learning allows a system of meaning to be discovered based on regularities inherent in the data. Unsupervised learning is possible in the absence of extra knowledge about the data, whereas supervised learning is not. Thus if our goal is to enable a system to be able to gather information from scratch, it seems unlikely that supervised learning alone will provide us a stable foundation to build on. Each supervised learning system will require another supervised learning system to establish the desired input-output relationship for the previous one, and the systems will chain in such a way forever.

Both supervised and unsupervised learning are thought to be present in some form in the brain, and connectionist models of the cortex have been created that incorporate both (O'Reilly and Munakata 2000).

Appendix B

Related Work

Other work towards establishing meaning for symbols has been conducted. (Agre and Chapman 1987) investigated the instantiation of general symbols using indexical-functional aspects. Aspects were intended to provide a meaning for objects in a world relative to their usefulness for an agent. Objects were instantiated as different aspects depending on how the agent could use them. This work was a significant step towards breaking free of a symbolic representation rooted only in meaningless symbols. Indexical-functional aspects added extra meaning into the system. Symbols could now be tracked in relation to the goals of the agent, which simplified the process of planning. Agre and Chapman's work falls short of creating the kind of representation discussed here, however, because 1) their functional representations are still rooted in the domain of playing the game, not in a general domain independent from it and 2) their representations are encoded in syntactic structures rather than real values.

(Drescher 1991) looked at how a system can build knowledge on top of knowledge through interaction in a simple micro-world with a hand, an eye, and objects. Much of Drescher's work is relevant, and in many ways this thesis is an attempt to improve on his results. The main limitation of the system he invented was the separation between his knowledge representation and his means of interacting with the world. His schemas were created in a theoretical space that did not impose much constraint on how data from the hand related to data from the eye. As a result, too many of his primitives were equivalent, and could fit into several places in his schemas.

Because of this and his departure from a more faithful model of sensation, Drescher also experienced some effects of the symbol grounding problem that we seek to avoid. Additionally, my system abstracts sensory data into clusters before associating them together. Drescher's system allowed all types of sensory information to associate so long as they fit into a schema.

(Beule, Looveren, and Zuidema 2002) approached the symbol grounding problem directly and built a blocks world in order to approach the problem. The system is given information about the objects such as their positions and their constituent properties. The system returns syntactic structures describing notable objects in a world such as "the red square moving to the right". While this produces interesting correlations, it is still a representation limited both by the hand-coded nature of the information given to it and the amount of generalizable knowledge that it can reuse. In contrast, my system operates on data that more plausibly resembles the kinds of data that the brain actually works with.

(Roy et al. 2003) demonstrates a particularly clear understanding of the need for symbol grounding. This work discusses a physically instantiated robot arm that can pick up objects. The distinction between this work and my own is that Roy uses both sensorimotor and linguistic primitives where I do not. Roy does not attempt to discover these sensorimotor primitives autonomously; they are hard-wired in. In contrast, my system attempts to answer the question of where those primitives come from, in addition to the question of how those primitives can become associated together.

Bibliography

Agre, P.E., and D. Chapman. 1987. "Pengi: An Implementation of a Theory of Activity." Edited by Morgan Kaufmann, *Proc. Sixth National ConferenceAmerican Association for Artificial Intelligence*, All ACM Conferences. American Association for Artificial Intelligence, 268–272.

Beule, Joachim De, Joris Van Looveren, and Willem Zuidema. 2002. "Grounding Formal Syntax in an Almost Real World." Ai-memo 02-03, Vrije Universiteit Brussel, Artificial Intelligence Laboratory.

Bickhard, Mark H., and Loren Terveen, eds. 1995. *Foundational Issues in Artificial Intelligence and Cognitive Science.* Advances In Psychology no. 109. North-Holland.

Brooks, Rodney A. 1991. "Intelligence without representation." *Artificial Intelligence*, no. 47:139–159.

Copeland, B.J. 2000, May. What is Artificial Intelligence? On the WWW at http://www.alanturing.net/turing_archive/pages/Reference%20Articles/what_is_AI/What%20is%20AI06.html.

Dittenbach, M., D. Merkl, and A. Rauber. 2000, July 24. – 27. "The Growing Hierarchical Self-Organizing Map." Edited by S. Amari, C. L. Giles, M. Gori, and V. Puri, *Proc of the International Joint Conference on Neural Networks (IJCNN 2000)*, Volume VI. Como, Italy: IEEE Computer Society, 15 – 19.

Drescher, Gary L. 1991. *Made-up Minds.* Cambridge, Massachusetts: MIT Press.

Fuster, Joaquin M. 2003. *Cortex and Mind: Unifying Cognition.* Oxford University Press.

Harnad, Stephen. 1990. "The Symbol Grounding Problem." *Physica D* 42:335–346.

Hertz, John A., Anders Krogh, and Richard G. Palmer. 1991, January. *Introduction to the Theory of Neural Computation.* Perseus Publishing.

Kohonen, Teuvo. 2001. *Self-Organizing Maps.* Third. Springer Series in Information Science no. 30. Springer.

Martin, Alex, and Linda L. Chao. 2001. "Semantic Memory and the brain: Structure and Processes." *Current Opinion in Neurobiology* 11 (2): 194–201 (April).

Minsky, Marvin. 1988. *The Society of Mind.* New York: Simon and Schuster.

O'Reilly, Randall C., and Yuko Munakata. 2000. *Computational Explorations in Cognitive Neuroscience.* Cambridge, Massachusetts: MIT Press.

Robinson, David A. 1992. "Implications of Neural Networks for How We Think About Brain Function." *Behav. Brain. Sci.* 15:644–55.

Romesburg, H. Charles. 1984. *Cluster Analysis for Researchers.* Belmont, California: Lifetime Learning Publications.

Roy, Deb, Kai-Yuh Hsiao, Nikolaos Mavridis, and Peter Gorniak. 2003. "Ripley, Hand Me The Cup! (Sensorimotor representations for grounding word meaning)." *Int. Conf. of Automatic Speech Recognition and Understanding.*

Winograd, Terry. 1971, Feb. "Procedures as a Representation for Data in a Computer Program for Understanding Natural Language." Mit ai technical report 235, MIT.

Winston, Patrick. H. 1993. *Artificial Intelligence.* Third. Reading, Massachusetts: Addison-Wesley.