Perception Processing for General Intelligence, Part II: Bridging the Symbolic/Subsymbolic Gap

Ben Goertzel

July 14, 2012

Abstract

Bridging the gap between symbolic and subsymbolic representations is a – perhaps *the* – key obstacle along the path from the present state of AI achievement to human-level artificial general intelligence. One approach to bridging this gap is hybridization – for instance, incorporation of a subsymbolic system and a symbolic system into a integrative cognitive architecture. Here we present a detailed design for an implementation of this approach, via integrating a version of the DeSTIN deep learning system into OpenCog, an integrative cognitive architecture including rich symbolic capabilities. This is a "tight" integration, in which the symbolic and subsymbolic aspects exert detailed real-time influence on each others' operations. Part I of this paper has described in detail the revisions to DeSTIN needed to support this integration, which are mainly along the lines of making it more "representationally transparent," so that its internal states are easier for OpenCog to understand. We describe an extension of the approach beyond vision to include multi-sensory integration, and perception-action integration. We discuss the potential use of this integrated system to control mobile robots, as exemplified via a thought-experiment involving eye-hand coordination.

1 Introduction

While it's widely accepted that human beings carry out both *symbolic* and *subsymbolic* processing, as integral parts of their general intelligence, the precise definition of "symbolic" versus "subsymbolic" is a subtle issue, which different AI researchers will approach in different ways depending on their differing overall perspectives on AI. Nevertheless, the intuitive meaning of the concepts is commonly understood:

- **"subsymbolic"** refers to things like pattern recognition in high-dimensional quantitative sensory data, and real-time coordination of multiple actuators taking multidimensional control signals
- **"symbolic"** refers to things like natural language grammar and (certain or uncertain) logical reasoning, that are naturally modeled in terms of manipulation of symbolic tokens in terms of particular (perhaps experientially learned) rules

Views on the relationship between these two aspects of intelligence in human and artificial cognition are quite diverse, including perspectives such as

1. Symbolic representation and reasoning are the core of human-level intelligence; subsymbolic aspects of intelligence are of secondary importance and can be thought of as pre or post processors to symbolic representation and reasoning
2. Subsymbolic representation and learning are the core of human intelligence; symbolic aspects of intelligence
   (a) emerge from the subsymbolic aspects as needed; or,
   (b) arise via a relatively simple, thin layer on top of subsymbolic intelligence, that merely applies subsymbolic intelligence in a slightly different way

3. Symbolic and subsymbolic aspects of intelligence are best considered as different subsystems, which
   (a) have a significant degree of independent operation, but also need to coordinate closely together; or,
   (b) operate largely separately and can be mostly considered as discrete modules

In evolutionary terms, it is clear that subsymbolic intelligence came first, and that most of the human brain is concerned with the subsymbolic intelligence that humans share with other animals. However, this observation doesn't have clear implications regarding the relationship between symbolic and subsymbolic intelligence in the context of everyday cognition.

In the history of the AI field, the symbolic/subsymbolic distinction was sometimes aligned with the dichotomy between logic-based and rule-based AI systems (on the symbolic side) and neural networks (on the subsymbolic side) [PJ88]. However, this dichotomy has become much blurrier in the last couple decades, with developments such as neural network models of language parsing [GH11] and logical reasoning [LBH10], and symbolic approaches to perception and action [SR04]. Integrative approaches have also become more common, with one of the major traditional symbolic AI systems, ACT-R, spawning a neural network version [LA93] with parallel structures and dynamics to the traditional explicitly symbolic version and a hybridization with a computational neuroscience model [JL08]; and another one, SOAR, incorporating perception processing components as separate modules [Lai12]. The field of "neural-symbolic computing" has emerged, covering the emergence of symbolic rules from neural networks, and the hybridization of neural networks with explicitly symbolic systems [HH07].

Our goal here is not to explore the numerous deep issues involved with the symbolic/subsymbolic dichotomy, but rather to describe the details of a particular approach to symbolic/subsymbolic integration, inspired by Perspective 3a in the above list: the consideration of symbolic and subsymbolic aspects of intelligence as different subsystems, which have a significant degree of independent operation, but also need to coordinate closely together. We believe this kind of integration can serve a key role in the quest to create human-level general intelligence. The approach presented here is at the beginning rather than end of its practical implementation; what we are describing here is the initial design intention of a project in progress, which is sure to be revised in some respects as implementation and testing proceed. We will focus mainly on the tight integration of a subsymbolic system enabling gray-scale vision processing into a cognitive architecture with significant symbolic aspects, and will then briefly explain how the same ideas can be used for color vision, and multi-sensory and perception-action integration.

The approach presented here begins with two separate AI systems, both currently implemented in open-source software:

- **OpenCog**, an integrative architecture for AGI [Goe10b] [GPW+11], which is centered on a "weighted, labeled hypergraph" knowledge representation called the Atomspace, and features a number of different, sophisticated cognitive algorithms acting on the Atomspace. Some of these cognitive algorithms are heavily symbolic in focus (e.g. a probabilistic logic engine); others are more subsymbolic in nature (e.g. a neural net like system for allocating attention and assigning credit). However, OpenCog in its current form cannot deal with high-dimensional perceptual input, nor with detailed real-time control of complex actuators. OpenCog is now being used to control intelligent characters in an experimental virtual world, where the perceptual inputs are the 3D coordinate locations of objects or small blocks; and the actions are movement commands like "step forward", "turn head to the right."

- **DeSTIN** [ARK09],[ARC09], a deep learning system consisting of a hierarchy of processing nodes, in which the nodes on higher levels correspond to larger regions of space-time, and each node carries out prediction regarding events in the space-time region to which it corresponds. Feedback and feedforward dynamics between nodes combine with the predictive activity within nodes, to create a complex nonlinear dynamical system whose state self-organizes to reflect the state of the world being perceived. The core concepts of DeSTIN are similar to those of Jeff Hawkins' Numenta system [HB06] [GH09], Dileep George's work (`http://vicariousinc.com`) and work by Mohamad Tarifi [TSH11], Bundzel and Hashimoto [BH10], and others. However, the specifics of DeSTIN's dynamics have been designed in what we consider a particularly powerful way, and the system has shown good results on small-scale

test problems [KAR10]. So far DeSTIN has been utilized only for vision processing, but a similar proprietary system has been used for auditory data as well; and DeSTIN was designed to work together with an accompanying action hierarchy.

These two systems were not originally designed to work together, but we will describe a method for achieving their tight integration via

1. Modifying DeSTIN in several ways, so that
   (a) the patterns in its states over time will have more easily recognizable regularities
   (b) its nodes are able to scan their inputs not only for simple statistical patterns (DeSTIN "centroids"), but also for patterns recognized by routines supplied to it by an external source (e.g. another AI system such as OpenCog)
2. Utilizing one of OpenCog's cognitive processes (the "Fishgram" frequent subhypergraph mining algorithm) to recognize patterns in sets of DeSTIN states, and then recording these patterns in OpenCog's Atomspace knowledge store
3. Utilizing OpenCog's other cognitive processes to abstract concepts and draw conclusions from the patterns recognized in DeSTIN states by Fishgram
4. Exporting the concepts and conclusions thus formed to DeSTIN, so that its nodes can explicitly scan for their presence in their inputs, thus allowing the results of symbolic cognition to explicitly guide subsymbolic perception
5. Creating an action hierarchy corresponding closely to DeSTIN's perceptual hierarchy, and also corresponding to the actuators of a particular robot. This allows action learning to be done via an optimization approach ([LKP+05], [YKL+04]), where the optimization algorithm uses DeSTIN states corresponding to perceived actuator states as part of its inputs.

The ideas described here have mostly not yet been implemented, but work has begun on Items 1a (modifying DeSTIN so that the patterns in its states over time will have more easily recognizable regularities) and 2 (utilizing Fishgram to recognize patterns in DeSTIN system states), as part of a 2012 Google Summer of Code project. Item 1a has been covered in the companion paper [Goeon]; the remainder of the points are discussed here.

2 OpenCog

For sake of brevity, we will review OpenCog only summarily here, directing the reader to appropriate references for more detail.

OpenCog is an open-source AGI software framework, which has been used for various practical applications, and also for the in-progress implementation of the OpenCogPrime design aimed ultimately toward AGI at the human level and beyond. OpenCog has been used for commercial applications in the area of natural language processing and data mining; e.g. see [GPPG06]. It has also been used to control virtual agents in virtual worlds, at first using an OpenCog variant called the OpenPetBrain [GEA08], and more recently in a more general way using a Minecraft-like virtual environment [GPC+11].

Conceptually founded on the "patternist" systems theory of intelligence outlined in [Goe06], OpenCogPrime combines multiple AI paradigms such as uncertain logic, computational linguistics, evolutionary program learning and connectionist attention allocation in a unified architecture. Cognitive processes embodying these different paradigms interoperate together on a common neural-symbolic knowledge store called the Atomspace. The interaction of these processes is designed to encourage the self-organizing emergence of high-level network structures in the Atomspace, including superposed hierarchical and heterarchical knowledge networks, and a self-model network enabling meta-knowledge and meta-learning.

OCP relies on multiple memory types (all intersecting via the AtomSpace, even when also involving specialized representations), including the declarative, procedural, sensory, and episodic memory types that are widely discussed in cognitive neuroscience [TC05], plus attentional memory for allocating system resources

generically, and intentional memory for allocating system resources in a goal-directed way. Declarative memory is addressed via probabilistic inference; procedural memory via probabilistic evolutionary program learning; episodic memory via simulation; intentional memory via a largely declarative goal system; attentional memory via an economics-based dynamical system similar to an attractor neural network. Sensorimotor memory is not handled thoroughly within OCP itself, part of the motivation for integrating DeSTIN into OpenCog as described in this paper.

The essence of the OCP design lies in the way the structures and processes associated with each type of memory are designed to work together in a closely coupled way, the operative hypothesis being that this will yield cooperative intelligence going beyond what could be achieved by an architecture merely containing the same structures and processes in separate "black boxes." That is, when a learning process concerned centrally with one type of memory encounters a situation where it learns very slowly, it can often resolve the issue by converting some of the relevant knowledge into a different type of memory: so-called **cognitive synergy**

OCP's dynamics has both goal-oriented and "spontaneous" aspects; here for simplicity's sake we will focus on the goal-oriented ones. The basic goal-oriented dynamic of the OCP system, within which the various types of memory are utilized, is driven by "cognitive schematics", which take the form

$$Context \wedge Procedure \rightarrow Goal < p >$$

(summarized $C \wedge P \rightarrow G$). Semi-formally, this implication may interpreted to mean: "If the context C appears to hold currently, then if I enact the procedure P, I can expect to achieve the goal G with certainty p." Cognitive synergy means that the learning processes corresponding to the different types of memory actively cooperate in figuring out what procedures will achieve the system's goals in the relevant contexts within its environment. In a robot or virtual "AGI school" context, for example, the top-level goals are simple things such as pleasing the teacher, learning new information and skills, and protecting the agent's body.

2.1 The Atomspace Representation

OpenCog's "Atomspace" knowledge representation is a generalized hypergraph formalism which comprises a specific vocabulary of Node and Link types, used to represent declarative knowledge and also, indirectly, other types of knowledge as well. There is a specific vocabulary of a couple dozen node and link types with semantics carefully chosen to reflect the needs of OpenCog's cognitive processes. Simple examples of OpenCog links, in the notation commonly used with OpenCog, are:

```
InheritanceLink Ben_Goertzel animal <.99>

EvaluationLink <.7>
        chase
        ListLink
                cat
                mouse
```

Examples using nodes with English-word labels provide convenient examples, but in fact most nodes in a practical OpenCog system will generally be automatically learned and not correspond directly to any human-language concept. We will see some examples involving sensorimotor data a little later, in the context of OpenCog-DeSTIN integration.

What's important about the AtomSpace knowledge representation is mainly that it provides a flexible means for compactly representing multiple relevant forms of knowledge, in a way that allows them to interoperate – where by "interoperate" we that e.g. a fragment of a chunk of declarative knowledge can link to a fragment of a chunk of attentional or procedural knowledge; or a chunk of knowledge in one category can overlap with a chunk of knowledge in another category (as when the same link has both a (declarative) truth value and an (attentional) importance value). In short, any representational infrastructure sufficiently flexible to support

- compact representation of all the key categories of knowledge playing dominant roles in human memory

- the flexible creation of specialized sub-representations for various particular subtypes of knowledge in all these categories, enabling compact and rapidly manipulable expression of knowledge of these subtypes
- the overlap and interlinkage of knowledge of various types, including that represented using specialized sub-representations

would probably be acceptable for OpenCog's purposes. The Atom formalism satisfies the relevant general requirements and has proved workable from a practical software perspective.

2.2 OpenCog's Cognitive Processes

.

The OpenCog cognitive processes most relevant to OpenCog-DeSTIN integration are as follows.

Fishgram and Sushigram , pattern recognition systems carrying out mining of frequent and surprising sub-hypergraphs from the Atomspace. Fishgram is somewhat similar to the GSpan [YH02] frequent subgraph mining algorithm, but modified to deal with hypergraphs as well as graphs, and also to utilize breadth-first rather than depth-first search, and to deal appropriately with mining patterns involving multiple variables. Most frequent subgraph mining algorithms use depth first search, because they have been customized for chemistry applications where this is appropriate; but given the statistics of OpenCog Atomspaces in practice, breadth-first search is more appropriate. Sushigram modifies Fishgram to search, not for the most frequent sub-hypergraphs of a hypergraph, but rather the most "surprising" in terms of interaction information [Bel03]. As a simple example, if an Atomspace contained many nodes representing characters that are both male and aggressive, Fishgram might identify a pattern of the form

```
ANDLink <.7>
        InheritanceLink $X male
        InheritanceLink $X aggressive
```

MOSES: Probabilistic Evolutionary Learning is a powerful framework for probabilistic evolutionary program learning [LG09]. It is used in OpenCog for a variety of purposes, among which the most relevant here is as an augmentation to Fishgram/Sushigram for mining Atomspace patterns. While Fishgram/Sushigram is a greedy approach in the tradition of Apriori and other frequent itemset miners, MOSES is an intelligent global search algorithm – slower than greedy methods at systematically finding all the simple patterns in a dataset, but able to find more complex patterns that greedy learning algorithms may miss.

OpenPsi [CGZ+11], an implementation of aspects of Dietrich Dorner's Psi model of human motivation and emotion [Bac09], structures OpenCog's goal-driven action selection. It is a specific implementation of the general "Context & Procedure → goal" approach mentioned above, mediating the process by which the system chooses which actions to take based on its various top-level and sub goals at the time and its perceptions and inferences regarding its situation.

Probabilistic Logic Networks or PLN [GIGH08] is OpenCog's probabilistic reasoning system, embodying mathematics allowing compact representations of probability distributions to be propagated through general logical inferences. Based on a unique integration of term logic and predicate logic, PLN allows commonsensical inference to be carried out silly and compactly. As an example, one of PLN's more basic inference rules (term logic deduction) is as follows:

```
InheritanceLink A B <sAB>
InheritanceLink B C <sBC>
|-
InheritanceLink A C <sAC>
```

where the formula

```
sAC = sAB sBC + (1-sAB) ( sC - sB sBC ) / (1- sB )
```

estimates the probability of the conclusion based on the probabilities of the premises. This rule would be used to draw conclusions such as

```
InheritanceLink object_456 has_head
InheritanceLink has_head can_converse
|-
InheritanceLink object_456 can_converse
```

which would be used by an OpenCog system controlling a game character to decide, on approaching an object with a head, that it may be worth trying to converse with that object. Most useful PLN inferences involve chaining together of multiple rules, and OpenCog contains PLN backward and forward chainers which construct such chains in appropriate ways. A core challenge here is scalable inference control, which is currently being addressed via a research project aimed at mining structures from prior inference chains to learn patterns tfor guiding future ones.

Economic Attention Allocation is OpenCog's framework for managing the memory and processing resources allocated to different Atoms. It manages the propagation of ShortTermImportance (STI) and LongTermImportance (LTI) values associated with Atoms. STI values are used to guide other cognitive processes in choosing which Atoms to deal with. LTI values are used to control forgetting; the Atoms with lowest LTI are removed from RAM. This component of the system also manages the formation of HebbianLinks between Atoms that are often simultaneously active, and also often simultaneously inactive; these links play a key role in OpenCog-DeSTIN integration, because it's relatively easy to mine them from DeSTIN state history, in relation to OpenCog Atoms representing aspects of DeSTIN states.

Concept Blending implements ideas from cognitive science [FT02] regarding the formation of new concepts from old, via combining pieces of the old in judicious ways. A standard example is formation of the concept of a "bat man" by putting together some aspects of "bat" (wings, flying, black color) with some aspects of "man" (legs, arms, head, intelligence). Many examples of scientific discovery and artistic creation have been shown analyzable in terms of blending. In an OpenCog-DeSTIN context, blending could lead to the creation of a new OpenCog ConceptNode representing a visual blend of a bat and a man, based on the visual patterns DeSTIN has formed based on observing bats and men.

2.3 Simplified OpenCog Workflow

The dynamics inside an OpenCog system may be highly complex, defying simple flowcharting, but from the point of view of OpenCog-DeSTIN integration, one important pattern of information flow through the system is as follows:

1. Perceptions come into the Atomspace. In the current OpenCog system, these are provided via a proxy to the game engine where the OpenCog controlled character interacts. In an OpenCog-DeSTIN hybrid, these will be provided via DeSTIN.
2. Hebbian learning builds HebbianLinks between perceptual Atoms representing percepts that have frequently co-occurred
3. PLN inference, concept blending and other methods act on these perceptual Atoms and their HebbianLinks, forming links between them and linking them to other Atoms stored in the Atomspace reflecting prior experience and generalizations therefrom
4. Attention allocation gives higher short and long term importance values to those Atoms that appear likely to be useful based on the links they have obtained
5. Based on the system's current goals and subgoals (the latter learned from the top-level goals using PLN), and the goal-related links in the Atomspace, the OpenPsi mechanism triggers the PLN-based planner, which chooses a series of high-level actions that are judged likely to help the system achieve its goals in the current context

6. The chosen high-level actions are transformed into series of lower-level, directly executable actions. In the current OpenCog system, this is done by a set of hand-coded rules based on the specific mechanics of the game engine where the OpenCog controlled character interacts. In an OpenCog-DeSTIN hybrid, the lower-level action sequence will be chosen by an optimization method acting based on the motor control and perceptual hierarchies.

This pattern of information flow omits numerous aspects of OpenCog cognitive dynamics, but gives the key parts of the picture in terms of the interaction of OpenCog cognition with perception and action. Most of the other aspects of the dynamics have to do with the interaction of multiple cognitive processes acting on the Atomspace, and the interaction between the Atomspace and several associated specialized memory stores, dealing with procedural, episodic, temporal and spatial aspects of knowledge. From the present point of view, these additional aspects may be viewed as part of Step 3 above, wrapped up in the phrase " and other methods act on these perceptual Atoms." However, it's worth noting that in order to act appropriately on perceptual Atoms, a lot of background cognition regarding more abstract conceptual Atoms (often generalized from previous perceptual Atoms) may be drawn on. This background inference incorporates both symbolic and subsymbolic aspects, but goes beyond the scope of the present paper, as its particulars do not impinge on the particulars of DeSTIN-OpenCog integration.

OpenCog also possesses a specialized facility for natural language comprehension and generation [LGE10] [Goe10a], which may be viewed as a parallel perception/action pathway, bypassing traditional human-like sense perception and dealing with text directly. Integrating OpenCog's current linguistics processes with DeSTIN-based auditory and visual processing is a deep and important topic, but one we will bypass here, for sake of brevity and because it's not our current research priority.

3 Integrating DeSTIN and OpenCog

The integration of DeSTIN and OpenCog involves two key aspects:

- recognition of patterns in sets of DeSTIN states, and exportation of these patterns into the OpenCog Atomspace
- use of OpenCog-created concepts within DeSTIN nodes, alongside statistically-derived "centroids"

From here on, unless specified otherwise, when we mention "DeSTIN" we will refer to "Uniform DeSTIN" as defined in the companion paper [Goeon], an extension of "classic DeSTIN" as defined in [ARK09].

3.1 Mining Patterns from DeSTIN States

The first step toward using OpenCog tools to mine patterns from sets of DeSTIN states, is to represent these states in Atom form in an appropriate way. A simple but workable approach, restricting attention for the moment to purely spatial patterns, is to use the six predicates:

- $hasCentroid(node\ N, int\ k)$
- $hasParentCentroid(node\ N, int\ k)$
- $hasNorthNeighborCentroid(node\ N, int\ k)$
- $hasSouthNeighborCentroid(node\ N, int\ k)$
- $hasEastNeighborCentroid(node\ N, int\ k)$
- $hasWestNeighborCentroid(node\ N, int\ k)$

For instance

$$hasNorthNeighborCentroid(N, 3)$$

means that N's north neighbor has centroid #3

One may consider also the predicates

- $hasParent(node\ N, Node\ M)$
- $hasNorthNeighbor(node\ N, Node\ M)$
- $hasSouthNeighbor(node\ N, Node\ M)$
- $hasEastNeighbor(node\ N, Node\ M)$
- $hasWestNeighbor(node\ N, Node\ M)$

Now suppose we have a stored set of DeSTIN states, saved from the application of DeSTIN to multiple different inputs. What we want to find are predicates P that are *conjunctions* of instances of the above 10 predicates, which occur frequently in the stored set of DeSTIN states. A simple example of such a predicate would be the conjunction of

- $hasNorthNeighbor(\$N, \$M)$
- $hasParentCentroid(\$N, 5)$
- $hasParentCentroid(\$M, 5)$
- $hasNorthNeighborCentroid(\$N, 6)$
- $hasWestNeighborCentroid(\$M, 4)$

This predicate could be evaluated at any pair of nodes $(\$N, \$M)$ on the same DeSTIN level. If it is true for atypically many of these pairs, then it's a "frequent pattern", and should be detected and stored.

OpenCog's pattern mining component, Fishgram, exists precisely for the purpose of mining this sort of conjunction from sets of relationships that are stored in the Atomspace. It may be applied to this problem as follows:

- Translate each DeSTIN state into a set of relationships drawn from: hasNorthNeighbor, hasSouthNeighbor, hasEastNeighbor, hasWestNeighbor, hasCentroid, hasParent
- Import these relationships, describing each DeSTIN state, into the OpenCog Atomspace
- Run pattern mining on this AtomSpace.

3.2 Probabilistic Inference on Mined Hypergraphs

Patterns mined from DeSTIN states can then be reasoned on by OpenCog's PLN inference engine, allowing analogy and generalization.

Suppose centroids 5 and 617 are estimated to be similar – either via DeSTIN's built-in similarity metric, or, more interestingly via OpenCog inference on the Atom representations of these centroids. As an example of the latter, consider: 5 could represent a person's nose and 617 could represent a rabbit's nose. In this case, DeSTIN might not judge the two centroids particularly similar on a purely visual level, but, OpenCog may know that the images corresponding to both of these centroids are are called "noses" (e.g. perhaps via noticing people indicate these images in association with the word "nose"), and may thus infer (using a simple chain of PLN inferences) that these centroids seem probabilistically similar.

If 5 and 617 are estimated to be similar, then a predicate like

```
ANDLink
   EvaluationLink
      hasNorthNeighbor
      ListLink $N  $M
   EvaluationLink
      hasParentCentroid
      ListLink $N  5
   EvaluationLink
```

```
      hasParentCentroid
      ListLink $M  5
   EvaluationLink
      hasNorthNeighborCentroid
      ListLink $N 6
   EvaluationLink
      hasWestNeighborCentroid
      ListLink $M 4
```

mined from DeSTIN states, could be extended via PLN analogical reasoning to

```
ANDLink
   EvaluationLink
      hasNorthNeighbor
      ListLink $N $M
   EvaluationLink
      hasParentCentroid
      ListLink $N  617
   EvaluationLink
      hasParentCentroid
      ListLink $M  617
   EvaluationLink
      hasNorthNeighborCentroid
      ListLink $N 6
   EvaluationLink
      hasWestNeighborCentroid
      ListLink $M 4
```

3.3 Insertion of OpenCog-Learned Predicates into DeSTIN's Pattern Library

Suppose one has used Fishgram, as described above, to recognize predicates embodying frequent or surprising patterns in a set of DeSTIN states or state-sequences. The next natural step is to add these frequent or surprising patterns to DeSTIN's pattern library, so that the pattern library contains not only classic DeSTIN centroids, but also these corresponding "image grammar" style patterns. Then, when a new input comes into a DeSTIN node, in addition to being compared to the centroids at the node, it can be fed as input to the predicates associated with the node.

What is the advantage of this approach, compared to DeSTIN without these predicates? The capability for more compact representation of a variety of spatial patterns. In many cases, a spatial pattern that would require a large number of DeSTIN centroids to represent, can be represented by a single, fairly compact predicate. It is an open question whether these sorts of predicates are really critical for human-like vision processing. However, our intuition is that they do have a role in human as well s machine vision. In essence, DeSTIN is based on a fancy version of nearest-neighbor search, applied in a clever way on multiple levels of a hierarchy, using context-savvy probabilities to bias the matching. But we suspect there are many visual patterns that are more compactly and intuitively represented using a more flexible language, such as OpenCog predicates formed by combining elementary predicates involving appropriate spatial and temporal relations.

For example, consider the archetypal spatial pattern of a face as: either two eyes that are next to each other, or sunglasses, above a nose, which is in turn above a mouth. (This is an oversimplified toy example, but we're positing it for illustration only. The same point applies to more complex and realistic patterns.) One could represent this in OpenCog's Atom language as something like:

```
AND
  InheritanceLink N B_nose
  InheritanceLink M B_mouth
  EvaluationLink
```

```
    above
    ListLink E N
EvaluationLink
   above
   ListLink N M
OR
  AND
     MemberLink E1  E
     MemberLink E2  E
     EvaluationLink
        next_to
        ListLink E1 E2
     InheritanceLink E1 B_eye
  AND
     InheritanceLink E  B_sunglasses
```

where e.g. *B_eye* is a DeSTIN belief that corresponds roughly to recognition of the spatial pattern of a human eye. To represent this using ordinary DeSTIN centroids, one couldn't represent the OR explicitly; instead one would need to split it into two different sets of centroids, corresponding to the eye case and the sunglasses case unless the DeSTIN pattern library contained a belief corresponding to "eyes or sunglasses." But the question then becomes: how would classic DeSTIN actually learn a belief like this? In the suggested architecture, pattern mining on the database of DeSTIN states is proposed as an algorithm for learning such beliefs.

This sort of predicate-enhanced DeSTIN will have advantages over the traditional version, only if the actual distribution of images observed by the system contains many (reasonably high probability) images modeled accurately by predicates involving disjunctions and/or negations as well as conjunctions. If the system's perceived world is simpler than this, then good old DeSTIN will work just as well, and the OpenCog-learned predicates are a needless complication.

Without these sorts of predicates, how might DeSTIN be extended to include beliefs like "eyes or sunglasses"? One way would be to couple DeSTIN with a reinforcement learning subsystem, that reinforced the creation of beliefs that were useful for the system as a whole. If reasoning in terms of faces (independent of whether they have eyes or sunglasses) got the system reward, presumably it could learn to form the concept "eyes or sunglasses." We believe this would also be a workable approach, but that given the strengths and weaknesses of contemporary computer hardware, the proposed DeSTIN-OpenCog approach will prove considerably simpler and more effective.

4 Multisensory Integration, and Perception-Action Integration

In Part I we have briefly indicated how DeSTIN could be extended beyond vision to handle other senses such as audition and touch. If one had multiple perception hierarchies corresponding to multiple senses, the easiest way to integrate them within an OpenCog context would be to use OpenCog as the communication nexus – representing DeSTIN centroids in the various modality-specific hierarchies as OpenCog Atoms (PerceptualCentroidNodes), and building HebbianLinks in OpenCog's Atomspace between these PerceptualCentroidNodes as appropriate based on their association. So for instance the sound of a person's footsteps would correspond to a certain belief (probability distribution over centroids) in the auditory DeSTIN network, and the sight of a person's feet stepping would correspond to a certain belief (probability distribution over centroids) in the visual DeSTIN network; and the OpenCog Atomspace would contain links between the sets of centroids assigned high weights between these two belief distributions. Importance spreading between these various PerceptualCentroidNodes would cause a dynamic wherein seeing feet stepping would bias the system to think it was hearing footsteps, and hearing footsteps would bias it to think it was seeing feet stepping.

And, suppose there are similarities between the belief distributions for the visual appearance of dogs, and the visual appearance of cats. Via the intermediary of the Atomspace, this would bias the auditory and

haptic DeSTIN hierarchies to assume a similarity between the auditory and haptic characteristics of dogs, and the analogous characteristics of cats. Because: PLN analogical reasoning would extrapolate from, e.g.

- HebbianLinks joining cat-related visual PerceptualCentroidNodes and dog-related visual PerceptualCentroidNodes
- HebbianLinks joining cat-related visual PerceptualCentroidNodes to cat-related haptic PerceptualCentroidNodes; and others joining dog-related visual PerceptualCentroidNodes to dog-related haptic PerceptualCentroidNodes

to yield HebbianLinks joining cat-related haptic PerceptualCentroidNodes and dog-related haptic PerceptualCentroidNodes. This sort of reasoning would then cause the system DeSTIN to, for example, upon touching a cat, vaguely expect to maybe hear dog-like things. This sort of simple analogical reasoning will be right sometimes and wrong sometimes – a cat walking sounds a fair bit like a dog walking, and cat and dog growls sound fairly similar, but a cat meowing doesn't sound that much like a dog barking. More refined inferences of the same basic sort may be used to get the details right as the system explores and understands the world more accurately.

4.1 Perception-Action Integration

While experimentation with DeSTIN has so far been restricted to perception processing, the system was designed from the beginning with robotics applications in mind, involving integration of perception with action and reinforcement learning. As OpenCog already handles reinforcement learning on a high level (via OpenPsi), our approach to robot control using DeSTIN and OpenCog involves creating a control hierarchy parallel to DeSTIN's perceptual hierarchy, and doing motor learning using optimization algorithms guided by reinforcement signals delivered from OpenPsi and incorporating DeSTIN perceptual states as part of their input information.

Our initial research goal, where action is concerned, is not to equal the best purely control-theoretic algorithms at fine-grained control of robots carrying out specialized tasks, but rather to achieve basic perception / control / cognition integration in the rough manner of a young human child. A two year old child is not particularly well coordinated, but is capable of coordinating actions involving multiple body parts using an integration of perception and action with unconscious and deliberative reasoning. Current robots, in some cases, can carry out specialized actions with great accuracy, but they lack this sort of integration, and thus generally have difficulty effectively carrying out actions in unforeseen environments and circumstances.

We will create an action hierarchy with nodes corresponding to different parts of the robot body, where e.g. the node corresponding to an arm would have child nodes corresponding to a shoulder, elbow, wrist and hand; and the node corresponding to a hand would have child nodes corresponding to the fingers of the hand; etc. Physical self-perception is then achieved by creating a DeSTIN "action-perception" hierarchy with nodes corresponding to the states of body components. In the simplest case this means the lowest-level nodes will correspond to individual servomotors, and their inputs will be numerical vectors characterizing servomotor states. If one is dealing with a robot endowed with haptic technology, e.g. Syntouch [FL12] fingertips, then numerical vectors characterizing haptic inputs may be used alongside these.

The configuration space of an action-perception node, corresponding to the degrees of freedom of the servomotors of the body part the node represents, may be approximated by a set of "centroid" vectors. When an action is learned by the optimization method used for this purpose, this involves movements of the servomotors corresponding to many different nodes, and thus creates a series of "configuration vectors" in each node. These configuration vector series may be subjected to online clustering, similar to percepts in a DeSTIN perceptual hierarchy. The result is a library of "codewords", corresponding to discrete trajectories of movement, associated with each node. The libraries may be shared by identical body parts (e.g. shared among legs, shared among fingers), but will be distinct otherwise. Each coordinated whole-body action thus results in a series of (node, centroid) pairs, which may be mined for patterns, similarly to the perception case.

The set of predicates needed to characterize states in this action-perception hierarchy is simpler than the one described for visual perception above; here one requires only

- $hasCentroid(node\ N, int\ k)$

- $hasParentCentroid(node\ N, int\ k)$
- $hasParent(node\ N, Node\ M)$
- $hasSibling(node\ N, Node\ M)$

and most of the patterns will involve specific nodes rather than node variables. The different nodes in a DeSTIN vision hierarchy are more interchangeable (in terms of their involvement in various patterns) than, say, a leg and a finger.

In a pure DeSTIN implementation, the visual and action-perception hierarchies would be directly linked. In the context of OpenCog integration, it is simplest to link the two via OpenCog, in a sense using cognition as a bridge between action and perception. It is unclear whether this strategy will be sufficient in the long run, but we believe it will be more than adequate for experimentation with robotic perceptual-motor coordination in a variety of everyday tasks. OpenCog's Hebbian learning process can be used to find common associations between action-perception states and visual-perception states, via mining a data store containing time-stamped state records from both hierarchies.

Importance spreading along the HebbianLinks learned in this way can then be used to bias the weights in the belief states of the nodes in both hierarchies. So, for example, the action-perception patterns related to clenching the fist, would be Hebbianly correlated with the visual-perception patterns related to seeing a clenched fist. When a clenched fist was perceived via servomotor data, importance spreading would increase the weighting of visual patterns corresponding to clenched fists, within the visual hierarchy. When a clenched fist was perceived via visual data, importance spreading would increase the weighting of servomotor data patterns corresponding to clenched fists, within the action-perception hierarchy.

4.2 Thought-Experiment: Eye-Hand Coordination

For example, how would DeSTIN-OpenCog integration as described here carry out a simple task of eye-hand coordination? Of course the details of such a feat, as actually achieved, would be too intricate to describe in a brief space, but it still is meaningful to describe the basic ideas. Consider the case of a robot picking up a block, in plain sight immediately in front of the robot, via pinching it between two fingers and then lifting it. In this case,

- The visual scene, including the block, is perceived by DeSTIN; and appropriate patterns in various DeSTIN nodes are formed
- Predicates corresponding to the distribution of patterns among DeSTIN nodes are activated and exported to the OpenCog Atomspace
- Recognition that a block is present is carried out, either by
  - PLN inference within OpenCog, drawing the conclusion that a block is present from the exported predicates, using ImplicationLinks comprising a working definition of a "block"
  - A predicate comprising the definition of "block", previously imported into DeSTIN from OpenCog and utilized within DeSTIN nodes as a basic pattern to be scanned for. This option would obtain only if the system had perceived many blocks in the past, justifying the automation of block recognition within the perceptual hierarchy.
- OpenCog, motivated by one of its higher-level goals, chooses "picking up the block" as subgoal. So it allocates effort to finding a procedure whose execution, in the current context, has a reasonable likelihood of achieving the goal of picking up the block. For instance, the goal could be curiosity (which might make the robot want to see what lies under the block), or the desire to please the agent's human teacher (in case the human teacher likes presents, and will reward the robot for giving it a block as a present), etc.
- OpenCog, based on its experience, uses PLN to reason that "grabbing the block" is a subgoal of "picking up the block"

- OpenCog utilizes a set of predicates corresponding to the desired state of "grabbing the the block" as a target for an optimization algorithm, designed to figure out a series of servomotor actions that will move the robot's body from the current state to the target state. This is a relatively straightforward control theory problem.
- Once the chosen series of servomotor actions has been executed, the robot has its fingers poised around the block, ready to pick it up. At this point, the action-perception hierarchy perceives what is happening in the fingers. If the block is really being grabbed properly, then the fingers are reporting some force, due to the feeling of grabbing the block (haptic input is another possibility and would be treated similarly, but we will leave that aside for now). Importance spreads from these action-perception patterns into the Atomspace, and back down into the visual perception hierarchy, stimulating concepts and percepts related to "something is being grabbed by the fingers."
- If the fingers aren't receiving enough force, because the agent is actually only poking the block with one finger and grabbing the air with another finder, then the "something is being grabbed by the fingers" stimulation doesn't happen, and the agent is less sure it's actually grabbing anything. In that case it may withdraw its hand a bit, so that it can more easily assess its hand's state visually, and try the optimization-based movement planning again.
- Once the robot estimates the goal of grabbing the block has been successfully achieved, it proceeds to the next sub-subgoal, and asks the action-sequence optimizer to find a sequence of movements that will likely cause the predicates corresponding to "hold the block up" to obtain. It then executes this movement series and picks the block up in the air.

This simple example is a far cry from the perceptual-motor coordination involved in doing embroidery, juggling or serving a tennis ball. But we believe it illustrates, in a simple way, the same basic cognitive structures and dynamics used in these more complex instances.

5 Conclusion

We have described, at a high level, a novel approach to bridging the symbolic / subsymbolic gap, via very tightly integrating DeSTIN with OpenCog. We don't claim that this is the only way to bridge the gap, but we do believe it is a viable way. Given the existing DeSTIN and OpenCog designs and codebases, the execution of the ideas outlined here seems to be relatively straightforward, falling closer to the category of "advanced development" than that of blue-sky research. However, fine-tuning all the details of the approach will surely require substantial effort.

While we have focused on robotics applications here, the basic ideas described could be implemented and evaluated in a variety of other contexts as well, for example the identification of objects and events in videos, or intelligent video summarization. Our interests are broad, however, we feel that robotics is the best place to start – partly due to a general intuition regarding the deep coupling between human-like intelligence and human-like embodiment; and partly due to a more specific intuition regarding the value of action for perception, as reflected in Heinz von Foerster's dictum "if you want to see, learn how to act". We suspect there are important cognitive reasons why perception in the human brain centrally involves premotor regions. The coupling of a perceptual deep learning hierarchy and a symbolic AI system doesn't intrinsically solve the combinatorial explosion problem intrinsic in looking for potential conceptual patterns in masses of perceptual data. However, a system with particular goals and the desire to act in such a way as to achieve them, possesses a very natural heuristic for pruning the space of possible perceptual/conceptual patterns. It allows the mind to focus in on those percepts and concepts that are useful for action. Of course, there are other ways besides integrating action to enforce effective pruning, but the integration of perception and action has a variety of desirable properties that might be difficult to emulate via other methods, such as the natural alignment of the hierarchical structures of action and reward with that of perception.

The outcome of any complex research project is difficult to foresee in detail. However, our intuition – based on our experience with OpenCog and DeSTIN, and our work with the mathematical and conceptual theories underlying these two systems – is that the hybridization of OpenCog and DeSTIN as described here will constitute a major step along the path to human-level AGI. It will enable the creation of an OpenCog

instance endowed with the capability of flexibly interacting with a rich stream of data from the everyday human world. This data will not only help OpenCog to guide a robot in carrying out everyday tasks, but will also provide raw material for OpenCog's cognitive processes to generalize from in various ways – e.g. to use as the basis for the formation of new concepts or analogical inferences.

References

[ARC09] I. Arel, D. Rose, and R. Coop. Destin: A scalable deep learning architecture with application to high-dimensional robust pattern recognition. *Proc. AAAI Workshop on Biologically Inspired Cognitive Architectures*, 2009.

[ARK09] I. Arel, D. Rose, and T. Karnowski. A deep learning architecture comprising homogeneous cortical circuits for scalable spatiotemporal pattern inference. *NIPS 2009 Workshop on Deep Learning for Speech Recognition and Related Applications*, 2009.

[Bac09] Joscha Bach. *Principles of Synthetic Intelligence*. Oxford University Press, 2009.

[Bel03] Anthony J. Bell. The co-information lattice. *Somewhere or other*, 2003.

[BH10] Bundzel and Hashimoto. Object identification in dynamic images based on the memory-prediction theory of brain function. *Journal of Intelligent Learning Systems and Applications*, 2-4, 2010.

[CGZ+11] Zhenhua Cai, Ben Goertzel, Changle Zhou, Yongfeng Zhang, Min JIang, and Gino Yu. Dynamics of a computational affective model inspired by drners psi theory. *Cognitive Systems Research*, 2011.

[FL12] Jeremy Fishel and Gerald Loeb. Bayesian exploration for intelligent identification of textures. *Frontiers in Neurorobotics 6-4*, 2012.

[FT02] G. Fauconnier and M. Turner. *The Way We Think: Conceptual Blending and the Mind's Hidden Complexities*. Basic, 2002.

[GEA08] Ben Goertzel and Cassio Pennachin Et Al. An integrative methodology for teaching embodied non-linguistic agents, applied to virtual animals in second life. In *Proc.of the First Conf. on AGI*. IOS Press, 2008.

[GH09] Dileep George and Jeff Hawkins. Towards a mathematical theory of cortical micro-circuits. *PLoS Comput Biol 5*, 2009.

[GH11] N Garg and J Henderson. Temporal restricted boltzmann machines for dependency parsing. In *Proc. ACL*, 2011.

[GIGH08] B. Goertzel, M. Ikle, I. Goertzel, and A. Heljakka. *Probabilistic Logic Networks*. Springer, 2008.

[Goe06] Ben Goertzel. *The Hidden Pattern*. Brown Walker, 2006.

[Goe10a] Ben et al Goertzel. A general intelligence oriented architecture for embodied natural language processing. In *Proc. of the Third Conf. on Artificial General Intelligence (AGI-10)*. Atlantis Press, 2010.

[Goe10b] Ben et al Goertzel. Opencogbot: An integrative architecture for embodied agi. *Proc. of ICAI-10, Beijing*, 2010.

[Goeon] Ben Goertzel. Perception processing for general intelligence, part i: Representationally transparent deep learning. (in preparation).

[GPC+11] Ben Goertzel, Joel Pitt, Zhenhua Cai, Jared Wigmore, Deheng Huang, Nil Geisweiller, Ruiting Lian, and Gino Yu. Integrative general intelligence for controlling game ai in a minecraft-like environment. In *Proc. of BICA 2011*, 2011.

[GPPG06] Ben Goertzel, Hugo Pinto, Cassio Pennachin, and Izabela Freire Goertzel. Using dependency parsing and probabilistic inference to extract relationships between genes, proteins and malignancies implicit among multiple biomedical research abstracts. In *Proc. of Bio-NLP 2006*, 2006.

[GPW+11] Ben Goertzel, Joel Pitt, Jared Wigmore, Nil Geisweiller, Zhenhua Cai, Ruiting Lian, Deheng Huang, and Gino Yu. Cognitive synergy between procedural and declarative learning in the control of animated and robotic agents using the opencogprime agi architecture. In *Proceedings of AAAI-11*, 2011.

[HB06] Jeff Hawkins and Sandra Blakeslee. *On Intelligence*. Brown Walker, 2006.

[HH07] Barbara Hammer and Pascal Hitzler, editors. *Perspectives of Neural-Symbolic Integration. Studies in Computational Intelligence, Vol. 77*. Springer, 2007.

[JL08] D. J. Jilk and C. Lebiere. and o'reilly. *R. C. and Anderson, J. R. (2008). SAL: An explicitly pluralistic cognitive architecture. Journal of Experimental and Theoretical Artificial Intelligence*, 20:197–218, 2008.

[KAR10] Tom Karnowski, Itamar Arel, and D. Rose. Deep spatiotemporal feature learning with application to image classification. In *The 9th International Conference on Machine Learning and Applications (ICMLA'10)*, 2010.

[LA93] C Lebiere and J R Anderson. A connectionist implementation of the act-r production system. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*, 1993.

[Lai12] John E Laird. *The Soar Cognitive Architecture*. MIT Press, 2012.

[LBH10] Jens Lehmann, Sebastian Bader, and Pascal Hitzler. Extracting reduced logic programs from artificial neural networks. *Applied Intelligence*, 2010.

[LG09] Moshe Looks and Ben Goertzel. Program representation for general intelligence. *Proc. of AGI-09*, 2009.

[LGE10] Ruiting Lian, Ben Goertzel, and Al Et. Language generation via glocal similarity matching. *Neurocomputing*, 2010.

[LKP+05] Sung Hee Lee, Junggon Kim, Frank Chongwoo Park, Munsang Kim, and James E. Bobrow. Newton-type algorithms for dynamics-based robot movement optimization. *IEEE Transactions on Robotics*, 21(4):657–667, 2005.

[PJ88] Steven Pinker and JacquesMehler. *Connections and Symbols*. MIT Press, 1988.

[SR04] Murray Shanahan and David A Randell. A logic-based formulation of active visual perception. In *Knowledge Representation*, 2004.

[TC05] Endel Tulving and R. Craik. *The Oxford Handbook of Memory*. Oxford U. Press, 2005.

[TSH11] Mohamad Tarifi, Meera Sitharam, and Jeffery Ho. Learning hierarchical sparse representations using iterative dictionary learning and dimension reduction. In *Proc. of BICA 2011*, 2011.

[YH02] X Yan and J Han. gspan: Graph-based substructure pattern mining. In *ICDM'02*, 2002.

[YKL+04] Sanghoon Yeo, Jinwook Kim, Sung Hee Lee, Frank Chongwoo Park, Wooram Park, Junggon Kim, Changbeom Park, and Intaeck Yeo. A modular object-oriented framework for hierarchical multi-resolution robot simulation. *Robotica*, 22(2):141–154, 2004.