The Knowledge Engineering Review, Vol. 26:2, 209–236.
doi:10.1017/S0269888911000038

Learning like a baby: a survey of artificial intelligence approaches

FRANK GUERIN

Department of Computing Science, University of Aberdeen, Aberdeen, Scotland;
e-mail: f.guerin@abdn.ac.uk

Abstract

One of the major stumbling blocks for artificial intelligence remains the commonsense knowledge problem. It is not clear how we could go about building a program which has all the commonsense knowledge of the average human adult. This has led to growing interest in the 'developmental' approach, which takes its inspiration from nature (especially the human infant) and attempts to build a program which could develop its own knowledge and abilities through interaction with the world. The challenge here is to find a learning program which can continuously build on what it knows, to reach increasingly sophisticated levels of knowledge. This survey reviews work in this area, with the emphasis on those that focus on early learning, for example, sensorimotor learning. The concluding discussion assesses the progress thus far and outlines some key problems which have yet to be addressed, and whose solution is essential to achieve the goals of the developmental approach.

1 Introduction

The idea of building an artificial baby dates back at least as far as Turing's paper on 'Computing Machinery and Intelligence' (Turing, 1950). He reckoned that it would be easier to write a program to simulate an infant's mind, rather than an adult's. The infant program could then be educated much like a human child, until it reached an adult level. The approach did not receive very much attention until Drescher's work of the late 80s (Drescher, 1991), which was itself an isolated work for a time. However, recent years have seen a rapid growth in the number of new attempts at what we shall call the *developmental approach* to artificial intelligence (AI). Many of these attempts are ongoing, producing new results each year. This is still a young area and holds much promise as a possible way to attack the hard problems of AI. This paper surveys the progress that has been made thus far.

Much of the work in developmental AI borrows ideas from developmental psychology, and conversely there are many psychological works which use computational models to explain their theories (Schlesinger & Parisi, 2001; Shultz, 2003). To define the scope of this paper, the first criterion has been that the work must be attempting to tackle AI problems by the developmental approach, rather than to illustrate a psychological theory for its own sake. By AI problems, we mean, for example, building programs that learn to make their own abstractions from sensor data, learn their own commonsense knowledge, or learn their own model of the environment they are situated in (rather than having these things specified by a human designer).

A second criterion that we applied is that the learning should be in a domain which bears some similarity to the natural physical world which we are familiar with, rather than some abstract domain where basic commonsense knowledge would not be needed. Therefore, we exclude programs which develop their knowledge of mathematics or Chess or Backgammon, for example. This excludes a number of works which display an interesting development of knowledge. For example, Colton

developed a mathematical discovery program (Colton, 2002), and Stracuzzi has developed a layered learner which has been applied to learning rules in the Solitaire card game (Stracuzzi, 2005), and Chess (Stracuzzi & Könik, 2008). These works build new concepts on what they have learnt previously and are therefore 'developmental'; however, we wish to keep the article focused on 'learning like a baby', with a view towards addressing the commonsense knowledge problem. It is possible that the techniques used in those works could be useful in learning basic commonsense knowledge at the sensorimotor level, but this has not been demonstrated. We provide some discussion of why we think it is necessary to focus on learning in realistic physical worlds at the end of Sections 4.1 and 4.2.

A third criterion is that we are focusing specifically on *early learning*. Thus, we will not address works which try to code the knowledge of a 3-year-old (for example), and then make the program learn more, in a developmental fashion[1]. A three-year-old has already acquired a great deal of commonsense knowledge, whereas we are focusing on programs which try to learn this knowledge, starting from a minimum level of innate knowledge. This focus on early learning also means that we do not include works which address the learning of language. We will call the works we focus on *early developmental AI*.

Notwithstanding the restrictions pertaining to AI problems, natural physical worlds, and early learning, this survey still has some overlap with surveys in epigenetic or developmental robotics (Lungarella *et al.*, 2003; Prince *et al.*, 2005), but can be further distinguished by the following three criteria. First, we will not restrict ourselves to physically embodied robots, but also include works which deal with simulated microworlds; in fact, most of the works reviewed here were evaluated by simulation. Second, we will not restrict ourselves to works that build on hypotheses put forward in developmental psychology or developmental neuroscience; we are interested in any work which attempts the developmental approach to AI, regardless of whether the work is faithful to developmental theories of biological organisms, or merely trying to build an AI program which develops knowledge, by whatever means. Third, we will exclude works on socially oriented interaction, because they tend to move away from our *early learning* criterion, especially those which deal with language. It is a moot point whether or not social interaction is a necessary component of early learning, but we exclude it anyway to limit the breadth of the review, and to devote more attention to the (non-social) learning of basic sensorimotor skills. This criterion alone makes this survey very different to Lungarella *et al.*'s (2003) as almost 40% of the papers reviewed by Lungarella *et al.* fell into the social learning category. Note that the first two criteria mentioned here remove the restrictions of the two criteria explicitly stated in the survey of Lungarella *et al.*

The paper is structured as follows: Section 2 gives the background to the developmental approach, explaining the motivation behind it, and its ultimate goals. Section 3 is the largest part of the paper and reviews existing works in early developmental AI, grouped according to themes. Section 4 assesses the progress that has been made thus far, and identifies some key problems which have yet to be tackled. Section 5 concludes.

2 Background

The motivation for the developmental approach to AI stems from the sheer difficulty of building an adult level AI. It is surmised that it may be easier to build an artificial baby, and let it learn. This is the idea outlined by Turing (1950); he estimated that the programming of an adult AI might take 60 programmers 50 years, and that it might be quicker to build a program to simulate the mind at birth, and then to educate it. Given that the history of AI has shown that the coding of an adult AI is considerably more difficult than Turing foresaw, it makes the developmental approach all the more attractive.

Quite apart from the programming time required for adult AI, there is the problem that we do not know how to design such a program. After over 50 years of AI research, some major stumbling blocks

[1] See, for example, the CogAff project `http://www.cs.bham.ac.uk/research/projects/cogaff/gc/`

are apparent, for example commonsense knowledge, representation, and generalization. These three are intimately related. We do not know the best representation to use to code commonsense knowledge; if we pick the wrong representation, then the facts we code in do not lend themselves to generalization; if we cannot generalize appropriately, then we need to code a great deal of commonsense knowledge to handle every special case. The developmental approach to AI is one possible way to attack these hard problems. It may be possible to build a baby program which can come up with representations which the designer had not conceived (Carey 1991, 1992, 2004). Thus, a baby program may be able to learn commonsense knowledge for itself. Generalization is perhaps the thorniest issue, but it is possible that if a baby learner can find the appropriate representations for its most basic concepts, generalizations may become easier. To some extent, the baby approach is proposing to tackle the generalization problem head on, because a developmental program will not get far if it cannot generalize well from its experiences.

In early AI research, there was a great effort to engineer systems with commonsense knowledge handcoded by human engineers. Although this approach proved very successful in tightly constrained domains, it did not work for general purpose systems requiring commonsense knowledge. Brooks (1991) criticized the approach of human engineers abstracting the world and coding high-level concepts in AI systems, for two reasons: first, the human engineer's conceptualization of the world may not be the appropriate one for an AI system with a different sensory and motor apparatus; and second, the abstraction which the human supposes to be appropriate may be completely different to what the human is actually using himself (because of the limits of introspection). Essentially, the criticism states that when human engineers try to make up the appropriate knowledge representation for an AI system, they will invariably do it wrongly. This has led to a trend towards avoiding the coding of high-level human knowledge, and instead approaching AI research from the bottom up; that is, by initially building robots which can deal with low-level sensorimotor data to interact with their world.

The developmental approach fits well with this recent trend in AI, and this partly explains why this area of research has been growing in popularity in recent years. Zlatev and Balkenius (2001) claim that large parts of the cognitive science community realize that 'true intelligence in natural and (possibly) artificial systems presupposes three crucial properties:

1. The embodiment of the system;
2. Its situatedness in a physical and social environment;
3. A prolonged epigenetic developmental process through which increasingly more complex cognitive structures emerge in the system as a result of interactions with the physical and social environment'.

Epigenetic here is Piaget's term (Piaget, 1971), meaning development in the individual through interaction with the environment. This belief has spawned the fields of *Epigenetic Robotics* and *Developmental Robotics*. Lungarella *et al*. (2003) explain the distinction between the two by saying that developmental robotics is more general than epigenetic robotics, and includes such things as the acquisition of motor skills and morphological development. The areas of epigenetic and developmental robotics span a rather broad range of research, much of which is concerned with modelling isolated aspects of intelligent behaviour, and many of which are solely concerned with cognitive modelling (rather than AI). Our interest here will be in those works which attempt to get to the bottom of the issue of building AI systems which can develop their own knowledge autonomously, and exhibit continuing development. This explains why we exclude works on later learning, and focus on early learning (especially basic sensorimotor development). Later learning must code in some commonsense knowledge (and decide on representations) which the system has not learnt autonomously, and so it goes against the pure developmental approach. We will focus on works which investigate how very basic sensorimotor knowledge could be learnt in the first place.

3 Approaches to early developmental artificial intelligence

In this section, we review the main existing works which have attempted the early developmental AI approach.

3.1 Piaget-inspired schemas

The works reviewed here draw their inspiration from Piaget's theory of sensorimotor intelligence (Piaget 1936, 1937, 1945). Central to this theory is the *schema*, which is a unit of knowledge. A schema gathers together some perceptions and (usually) associated actions. A typical example could be the schema of pulling a string to shake a rattle. This schema includes visual and tactile perceptions of the string, the action of grasping and pulling, and the expectation of hearing and seeing the rattle shake. Piaget describes how the infant is born with some innate reflex schemas which include, for example, a schema for sucking to feed, or looking towards light. Schemas always seek opportunities to repeat themselves, and in doing so they generalize to new situations, differentiate into new schemas, combine with other schemas, etc. Through these processes, the reflex schemas create more sophisticated schemas, which themselves go on to create more sophisticated schemas, and this explains how the infant exhibits increasingly sophisticated forms of behaviour. Piaget traces infant cognitive development from birth, and gives an account of how each new behaviour in the infant's repertoire can be explained by the operation of the previously existing schemas.

3.1.1 Drescher's simulated infant

The doctoral thesis of Gary Drescher (1991) presented a computational model of Piaget's schema mechanism. Drescher simulated a baby, with a hand, eye, and mouth, in a 7×7 grid world. The world also contained some objects which the baby could grab. Drescher's schemas were three-part structures consisting of a context, action, and result. A schema is a prediction about the world: if its action is taken in the context specified, then the result is predicted. For example, one schema which the program learnt is that if its current context was 'HandInFrontOfMouth', and it took the action 'HandBackwards', then it would expect to obtain the result 'HandTouchingMouth'. Schemas maintain (and constantly update) a record of the reliability of their prediction.

A new schema starts out *bare*, meaning that it is only an action, and does not know its context or result. It then learns the possible results it can achieve by monitoring what follows its activation, over a number of trials. Context conditions can be added to a schema if they increase the reliability of its result prediction. Drescher's implementation has been described as inordinately inefficient (Witkowski, 1997), and certainly would not scale up to larger worlds. Chaput (2004) elaborates on some specific problems: whenever a new bare schema is created, or a new result is found for a bare schema, or a new context is found, all of these represent new individual schemas which must be maintained, and which must be checked (to update their statistics) on every iteration of the learner. The mechanism only adds one result or context element at a time to a schema, and keeps all the intermediate schemas generated on the way to finding a complex reliable schema. Despite these efficiency problems, the program learnt a number of schemas which mimicked some of the acquisitions described by Piaget. After exploration, it was able to reliably predict the effects of most of its actions from whatever context it was in.

Drescher also included a pair of objects in the world, which the eye could see, and the hand could touch and grab. However, the objects moved occasionally of their own accord; thus, the schemas for grabbing them were not entirely reliable. Drescher introduced the idea of the 'synthetic item' to cope with this; the synthetic item could be 'on' if grabbing had worked recently, and thus could be used to predict if grabbing the object was likely to work. As soon as grabbing ceased to work (because the object had moved), the synthetic item would be switched off, thus predicting that further grabs would not work. The 'synthetic item' is interesting because the program is starting to learn higher-order data which goes beyond what is directly sensed. This is in effect an abstraction of the raw sensor data which allows predictions to be made about objects in the world, and so it arrives at an extensional approximation of the concept of an object (i.e. it is generally 'on' when the object is present).

Drescher's work has been quite influential, and has been frequently cited subsequently. We will see that many of the works reviewed in subsequent subsections follow the basic idea of 3-part

schemas which update their conditions for success (context) and result; thus, we can conclude that the approach has significant generality. The synthetic item is perhaps the most noteworthy aspect of Drescher's work, which has inspired other works seeking to learn about hidden states. Morrison *et al.* (2001) describe how an agent could construct new predicates, in a similar way to Drescher's synthetic item creation, for example, to represent hidden attributes such as the weight of an object. Overall, Drescher's work is a promising approach to developmental AI, but the world used is too simple to allow it to get as far as learning commonsense knowledge, and the efficiency problems preclude its use in a more complex world. Nevertheless, the basic ideas of learning context–action–result triples, and synthetic items, are very promising, and could well be used profitably in a more efficient implementation.

An interesting recent work by Holmes and Isbell (2005) generalizes and improves on Drescher's mechanism, and views the problem as one of learning a POMDP (partially observable Markov decision process). Holmes and Isbell note that the synthetic item is a means of implicitly summarizing the preceding history without explicitly remembering it; this history is useful to disambiguate between states which otherwise appear identical (i.e. it is a way to overcome the 'state aliasing problem'). In a subsequent work, Holmes and Isbell (2006) have shown that looping 'prediction suffix trees' (PSTs) can represent deterministic POMDPs. Once a looping PST is constructed for a deterministic POMDP, then we can use an observed history to trace a path through the tree and find out what state we are in the underlying MDP (Markov decision process), and hence predict what can happen next. They have shown that there is a finite looping PST for any finite, deterministic POMDP, which can be used to predict perfectly. They have also given an efficient algorithm for learning a looping PST from a sequence of sensorimotor experience. To connect back to Drescher's work: whenever Drescher's learning mechanism would need to create an extra synthetic item in order to predict accurately, this simply translates to an increase in the depth of the corresponding looping PST.

3.1.2 Chaput's model of Cohen's principles

Chaput's doctoral thesis (Chaput, 2004) recreated the achievements of Drescher, and went significantly further. Chaput developed a 'constructivist learning architecture' (CLA) which is based on Leslie Cohen's theory of infant cognitive development Cohen (1998); this is a neo-Piagetian theory which provides a little more detail than Piaget did about the required information processing mechanism. Cohen has abstracted, from many studies on infants, a set of information processing principles which apply throughout development and across all domains. These principles state that infants learn to process information at increasingly higher levels of abstraction by forming higher-level units out of relationships among lower-level units. There is a bias to process information using the highest formed units, unless the input becomes too complex, in which case the infant drops back to a lower level and attempts to refine its abstraction so as to be able to handle the complex information at the higher level. Essentially, Cohen's principles describe a strategy for making abstractions; from the masses of raw data, the infant need only pay attention to the abstractions it has found useful (i.e. it searches for high-level relationships among the highest level of abstractions created thus far, rather than looking at all the lower levels each time).

Chaput's computational model (CLA) is based on self-organizing maps which are built hierarchically (modelling the different levels of Cohen's theory). Chaput demonstrated that the CLA successfully models some aspects of infant development, in particular the perception of causality. He then applied it to Drescher's microworld, and to a robot learning task. The CLA implementation of Drescher's schema mechanism brought a number of efficiency improvements; for example, the CLA provided a finite data space for schemas (and synthetic items) to be learnt, so that their number could not grow exponentially as in Drescher's implementation. Chaput provided a process for different representations to compete for the finite available space. We noted above that Drescher's mechanism only adds one result or context element at a time to a schema, and keeps all the intermediate schemas generated along the way to a complex schema; the CLA, by contrast, can go straight to a complex schema without the intermediates. Furthermore, the CLA

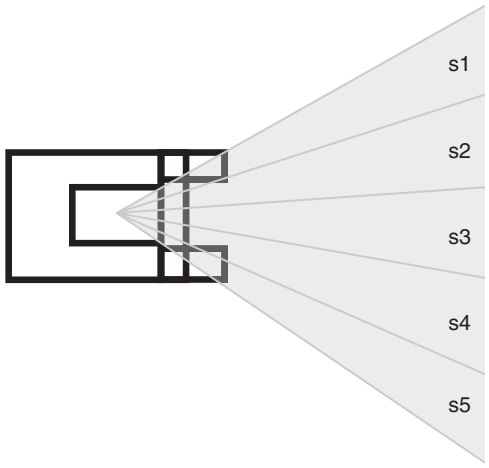

Figure 1 The five sectors in the visual system of Chaput's forager

learns schemas at a certain level first until it has some stable knowledge, before moving on to consider synthetic items at the next level. Once the CLA has moved to a higher level, those schemas from the lower level are not updated anymore; learning at that level has frozen. Thus, the learning resources can focus on one level at a time. The CLA successfully replicated all the functionality of Drescher's mechanism.

In Chaput's robot learning task, the robot had to 'forage' (i.e. see objects, move towards them, and pick them up). The robot had a vision system which had a 60° viewing angle, separated into five sectors (s1 … s5) of 12° each (see Figure 1). When a blob appeared in one of these sectors, a binary sensor item was set to true. In the foraging task, Chaput's system came up with many interesting and very useful synthetic items, which led to very efficient performance on the task. For example, if the robot sees an object in sector s2, and wants to pick it up, the robot should first turn left (to move the object into sector s3) and then move forward to pick it up. Now, usually one would expect that if an object is in the centre of vision (sector s3), and the robot moves forward, then the object should remain in the centre. However, this does not hold if the object was not exactly in the centre; it may drift to one side as we move forward. In fact, this tended to happen if the object had been in s2 and was moved to s3 by a left turn. The system learnt to recognize this situation of the object not being exactly in the centre via synthetic items. The system then learnt how to do a double turn at the start and then move straight to the specimen, rather than going forward and having to readjust its orientation in the middle. Thus, we can see how the CLA is able to compensate for its impoverished sensory apparatus; it is effectively learning how to represent states of the world which are not directly observable to its sensors in one time step (e.g. the object is at one side of a sensor region). This is quite impressive for a learner which was given no special prior knowledge about its world.

Overall, the CLA is a very promising architecture for developmental AI. The fact that candidate schemas compete for a limited space (via self-organizing maps) ensures the scalability of the system. The architecture also has great generality as the designer can choose to build higher layers (in the hierarchy) from any desired combination of lower layers; for example, integrating multi-modal information from lower layers, or integrating time-delayed versions of sensory input.

3.1.3 Stojanov's Petitagé

Stojanov *et al.*, in a number of papers, have been pursuing a Piagetian approach using a mechanism which is similar to the schema learning of Drescher and Chaput. Stojanov *et al.* (1997) broadly agree with Brooks's view that to design intelligent agents it will be necessary to create a complete creature that can survive in a dynamic environment, rather than the traditional approach of studying and modelling isolated aspects of intelligence. However, Stojanov *et al.* feel that

Brooks and his followers have placed too much emphasis on reactivity, and neglected to tackle the important issues of internal representation and learning. To tackle these issues, Stojanov *et al.* draw inspiration from Piaget's theory, whereby an agent constructs its own representation of reality through its interactions. In this way, it does not fall foul of Brooks's criticism of classical AI, which uses the human designer's conception of the world (as discussed in Section 2).

Stojanov *et al.* (1997) have designed an agent called Petitagé which inhabits a world with walls which it must avoid, and food which it must find. The agent learns 'expectancies' through interacting with the world. These expectancies are like Drescher's schemas in that they have a $\langle$ *Sensor_state*, *action*, *Sensor_state* $\rangle$ structure, meaning the state before the action, the action, and the state after. By recording the expectancies it learns, the agent builds a partial map of its world (i.e. the map is represented by a network of expectancies). This is how the agent *constructs* its world model, and *assimilates* the environment, and so is in line with Piaget's theory. Expectancies also have an emotional context which gives the expectancy a value describing how good it is at satisfying a particular drive (such as hunger). If the expectancy has a high value, then it should lead to satisfaction soon. The learning can be seen as a type of reinforcement learning, because when a drive gets satisfied, then the emotional contexts on the path which led to that satisfaction are updated backwards along that path.

Stojanov (2001) extends the above work by introducing Piagetian *schemas*. In this work, a schema is a sequence of actions, such as 'forward, forward, left, forward, right'. When the agent executes a schema, only some of the actions are achievable, because of walls being in the way. Thus, the above schema may degenerate to 'forward, forward, forward' when executed in a long corridor. This is called an *enabled schema action sequence*. The agent learns links between pairs of enabled schema action sequences which can happen consecutively; thus, if the agent is in a situation where the first schema has been matched, then it expects to be able to carry out the actions in the second schema. Links are stored in an associative memory. Goal-directed behaviour can be implemented by looking at the associative memory to find the link which has the shortest distance to the desired goal.

Using this mechanism, the agent learns cycles, where a cycle connects a chain of schemas back to its starting point. Cycles are intended to model higher-order knowledge of the structure of the environment. If the agent is solving a navigation task, the cycles can be thought of as places (Kulakov & Stojanov, 2002). The agent has been given a mechanism for detecting cycles, and can also find actions to move between the different cycles that have been learnt. Kulakov and Stojanov (2002) also describe how the process of finding higher-order regularities could be continued, finding regularities in the regularities, etc. This would result in the agent finding representations which are increasingly more detached from raw sensory input. It is hoped that such representations could 'eventually be used as manipulable protosymbols in the processes of hypothesis generation (about unknown environments) or communication with other same type agents'.

In Stojanov and Kulakov (2003), the Petitagé work was extended by giving the agent the ability to distinguish between different environments; one was a closed maze-type environment with corridors, and the other an open environment with some obstacles. The agent learnt schemas for each, and then a metric was created to allow the similarity between schemas to be quantified. It was then found that the schemas in similar environments clustered together, and thus the agent could use this to determine what type of environment it was in.

Overall, the work of Stojanov *et al.* brings a number of new innovations to the ideas of Drescher's schema mechanism; however, it is difficult to evaluate their relative merits when they are not applied to the same domains as the works previously reviewed. For example, Stojanov introduces *drives* such as hunger, whereas Drescher and Chaput merely have schemas which desire to repeat themselves for the sake of repetition. Furthermore, both have different approaches to learning higher-order knowledge, and it would be interesting to see exactly what they could learn if put in the same world. The relative merits of each approach are not clear. This points to a need more generally to have some standard platforms and worlds in which developmental AI programs could be tested.

3.1.4 The constructivist anticipatory learning mechanism system

Perotto and Álvares (2006)[2] also have a tripartite schema structure consisting of context, action, and expectation (just like Drescher's). As the agent learns more about its environment, it modifies its schemas by one of the following three methods: differentiate, adjust, or integrate. Differentiation can happen when a general schema produces an unexpected result, and leads to the creation of two new more specific schemas. Adjustment happens when an erroneous prediction is made and differentiation is not possible (because a schema is already specific). In this case, any erroneous element in the expectation is simply generalized. The final method of modification, integration, is simply differentiation in reverse, where two specific schemas with the same expectation can be replaced by a single schema with a more general context.

These methods could be beneficial for other Piagetian systems such as Chaput's. Recall that Chaput's system freezes schemas once one layer is learnt, so no further modification could happen to these schemas. Of course, freezing could easily be suspended if desired, but even then Chaput's system could benefit from the 'integrate' method. Chaput's learner can sometimes come up with many schemas in which all predict the same result, but have various different contexts, which could be generalized to a single schema with no loss of predictive power.

Perotto *et al.* (2007) extended this work to include 'synthetic elements' much like Drescher's synthetic item. Synthetic elements are created to model unobserved aspects of the environment which could be used to disambiguate between two schemas which have the same context, and hence to accurately determine the appropriate expectation. These synthetic elements are effectively performing the same function as displayed by Chaput's forager when it learnt to represent unobserved aspects of the environment. Recall that Chaput's forager would create a synthetic item for an unreliable schema, and then connect this back to a schema which causes it to happen reliably. However, Chaput's synthetic items are not restricted to being connected back to those schemas which 'switch them on'; instead, they are available to be treated in a similar way to sensor data, and to be correlated with any other schemas. This gives Chaput's system greater generality.

3.2 Cohen's image schemas

A quite different approach to schema learning has been pursued by Paul Cohen *et al.* over a number of years. We review these works roughly in order of publication. Their initial work was based on the idea of 'image schemas', inspired by the work of Mandler (1992) and Lakoff and Johnson (1980). They wish to show that an artificial infant could be developed to learn the kind of primitive interactional knowledge which could serve as the grounding for adult concepts (following Lakoff and Johnson's ideas). Cohen *et al.* (1997), by their own admission, are 'anti-nativist' and 'minimalist' in their approach. They aim to show that an agent can develop sophisticated knowledge of the world from very minimal beginnings.

In the first work in this series Cohen *et al.* (1997), they aim to show that concepts and categories can be learnt through interaction with the environment. They describe a simulated baby called Neo which lives in 'BabyWorld', where he can move his arm and head, and grasp several objects; for example, he can grasp rattles which make noise when shaken. Neo learns in a hierarchical fashion, beginning with low-level 'streams' of data (which describe things such as shape and colour of seen objects), and building up to higher level knowledge such as that having grabbed the wooden rattle, he can begin sucking it. By abstracting from similar chains of actions, the authors have noted that Neo has implicitly learnt classes of objects. For example, if Neo discovers that he can grab and suck both plastic and wooden objects, then he has implicitly learnt that these objects can be put in the same class. Thus, Cohen *et al.*'s original goal is achieved to some extent, in that Neo can learn some concepts in terms of the interactional properties of objects. In the longer term, Cohen *et al.* believe that their method of learning could be used to learn concepts such as

[2] Their agent operates in an abstract domain, and so the work is not quite within the scope of this review; however, the work is so close technically that it is worth including for purposes of comparison.

'containment'; this is an example of one of the concepts described by Lakoff and Johnson (1980), being initially learnt for physical objects, but then applied to many different abstract ideas, such as containing a thought in one's head.

To compare this work with the other developmental AI works is again (as in Stojanov's case) difficult, because of the different simulated worlds which they deal with. The statistical methods used are similar to Drescher's, but again, finding correlations among all sensor streams is only feasible for small domains; additional techniques would need to be introduced to scale up to more complex domains where there are many more things to pay attention to. Nevertheless, this does not necessarily detract from the usefulness of the hierarchical learning methods in a developmental AI system. One aspect is missing from the work in that Cohen *et al*. are dealing with sensor data which are to some extent already abstracted, as they receive symbols describing shape and colour ready made. To connect the system to lower-level sensor data would require an additional abstraction layer, where meaningful concepts such as shape and colour are learnt. This lower-level learning is one of the thorniest issues in AI. The work then falls foul of Brooks's criticism of abstraction. A question arises as to whether it is necessary to connect with low-level sensors to make a useful developmental AI system; we conclude that it is necessary in our further discussion which is given in Section 4.2.

A related work which attempts to learn image schemas is that of Rosenstein *et al*. (1997). This work focuses on learning image schemas for dynamic concepts such as one agent *avoiding* another, or one agent attempting to *crash* into another. To recognize these types of behaviour, Rosenstein *et al*. (1997) built *behaviour maps* for each type of behaviour. Let us take the behaviour map of *avoid* as an example: given the proximity of the agent to another, and the rate at which they are getting closer, the behaviour map describes how strongly the agent attempts to get away. Having built a library of behaviour maps, for nine different behaviours, it was then possible to take an observed trajectory of an agent and recognize which behaviour was being displayed by the agent. The recognition was reasonably accurate, although some confusion occurred between similar behaviours. The authors then went further to learn maps to describe the interaction between pairs of behaviours. The resulting interaction maps not only allow pairs of behaviours to be recognized, but also allow a prediction to be made of whether the observed agents are likely to make contact. This ability to make predictions based on what has been recognized strengthens the authors' case that interaction maps are in fact concepts. In a subsequent work, Rosenstein and Cohen (1998) improved the accuracy of recognition and prediction using an 'attractor reconstruction' technique.

This work addresses the criticism we made of Cohen *et al*. (1997) above, because Rosenstein *et al*. (1997) learns a concept from scratch. However, there are a number of questions relating to its generality which would need to be answered by further research. It is not clear how it could be applied to learning things other than behaviour, for example, object concepts, or relationships among objects. It would also need to be tested on more complex domains (e.g. worlds with more agents and more sensor data) in order to determine how scalable it is; behaviour maps would be multi-dimensional in this case.

In a more recent work, Amant *et al*. (2006) describe a language for image schemas which allows three types of schemas to be represented: static schemas, dynamic schemas, and action schemas. Static schemas can capture such things as *containment* relationships among objects. Dynamic schemas can capture dynamic relationships such as one object *approaching* another. Action schemas can be made up by chaining together dynamic schemas; for example, *A* 'kicking' the ball *B* could be a sequence of three dynamic states: *A* approaching *B*, *A* coming in contact with *B*, and *A* stopped with B moving away. The authors explain how the use of image schemas holds the promise of getting away from rigid state descriptions of a world, and moving towards more flexible models which could transfer to different domains. In their example, a learnt sequence of image schemas could describe sneaking up on a cat (a slow approach) and quickly pouncing on it (a fast approach) to catch it. Such compositions of schemas are called *gists*. These gists can capture the essential relationships among the objects and could potentially be transferred to other domains. They also mention the possibility of *specializing* schemas; for example, 'push' and 'shove'

schemas could be derived from an 'apply force' schema; this corresponds to the Piagetian idea of *differentiating* schemas.

Building on these ideas, the 'Jean' system Chang *et al.* (2006) combines the use of image schemas and gists with a number of Piaget's ideas, for example, (i) that schemas should repeat themselves as that is intrinsically rewarding, (ii) that a 'relatively small, core set of schemas and a general algorithm' are all that is needed to progress to higher levels of knowledge, and (iii) that 'newly learned schemas are assembled from previously learned and appropriately modified components'. The ESS (experimental state splitting) algorithm is an important part of the Jean system. Jean models the state of the world by using a simple model, typically starting with only one or two states. It tries to predict what state will result from each action that it can take. For some state/action pairs, Jean is very uncertain about which state might result; these states are selected for splitting. This can be illustrated with an example. As in the example from the previous paragraph, the Jean system attempts to catch a cat. Initially, the system has only two states to model the world; one is its goal state of contacting the cat, and the other is the non-goal state. As it attempts to catch objects, it notes that balls are easier to catch than cats, and so splits its non-goal state into one for cats and one for balls. The cat state is split further when Jean realizes that being within a certain distance of the cat is a good predictor of whether or not a catching attempt will be successful. In this way, Jean can learn gists (combinations of image schemas) that can achieve its goal. The authors also experimented with transfer of learning, between the tasks of catching the cat and catching the ball, and showed that prior learning did bring a benefit.

Cohen *et al.* (2007) give a little more detail on the nature of Jean's 'action schemas': they have three components: controllers, maps, and decision regions. Controllers allows Jean to control its behaviour, for example, to move; this leads to a change in some of the variables sensed by Jean, for example, distance to an object, or velocity. The 'maps' can represent a space with dimensions of these changing variables (e.g. distance and velocity); therefore, the maps can record the activation of a schema as a trajectory in this space. Finally, decision regions bound certain areas of the space in a map where it is appropriate for Jean to switch from one controller to another. For example, if trying to approach an object within a certain time, there is a region on the corresponding map which means that from Jean's current position it will be impossible to get there in time.

The Jean system is one of the most impressive we have reviewed thus far in that it can deal with sensors over a continuous range, and can find critical thresholds to split states and discover regions where different behaviours are found. In this respect, it addresses some of our concerns about the earlier works, because it is finding its own abstractions for the low-level data autonomously. Nevertheless, to take the learning further, we expect that it would be necessary for the learner to have access to much more low-level sensor data. For example, Jean has access to the distance between itself and the cat from the beginning. This is an advantage, however, as it is very helpful in order to catch the cat in this scenario. However, since it has is a given, the system cannot move to a lower level and find a new relationship among raw positions of objects, for example, which might be needed to solve a slightly different problem. This relates to Sutton's 'Verification Principle', which is discussed further in Section 4.2.

3.3 Intrinsic motivation

Oudeyer *et al.* (2007) tackle the problem of creating an agent which will go through a development sequence, moving towards more complex stages of behaviour, motivated only by its own intrinsic curiosity. This is called the mechanism of 'Intelligent Adaptive Curiosity'. This was demonstrated with a Sony Aibo which was situated on a 'play mat' where there was a piece of material which could be bitten, and a hanging object which could be bashed. The Aibo had various primitive motor actions such as pan and tilt of the head, bashing strength and bashing angle, and biting. Its high-level sensors could detect if the robot sees an object, if the robot is biting an object, and if the robot sees something oscillating (e.g. the hanging object swinging). The robot did display a sequence of increasingly complex behaviours, where some built on earlier ones.

The robot maintains predictive schemas similar to Drescher's, and the learning is driven by an 'internal' reward which is proportional to the decrease in its error rate in prediction. The learning mechanism used is a simple type of reinforcement learning. The agent chooses actions to maximize its reward, and therefore it chooses actions which maximize its learning progress. An essential part of the mechanism is the idea of *regions*; a region stores schemas as the agent learns. When the agent has recorded 250 schemas in one region, it splits the region in two, in a way which minimizes the sum of the variances of the schemas in each new region. Each new region then has its own learning machine devoted to it. This region splitting leads to 'stages' of behaviour, because the agent may find that after doing a split, the actions in one region may lead to a sharp decrease in error rate (hence, a higher reward), in comparison to the other region; this means that the agent will favour action selection in the first region. In the robot scenario, this leads the robot to focus on different behaviours at different times; for example, he focuses on just looking around for a time, followed by a focus on biting actions, followed by a focus on bashing, etc.

The ordering of Oudeyer *et al.*'s various stages is one of increasing complexity; looking is the simplest behaviour, controlled by the pan and tilt motor primitive, while biting requires one more motor primitive, and bashing must also consider strength and angle. More interestingly, from our developmental point of view, there is also evidence of skills building on each other; the experiments showed that non-affordant biting precedes affordant biting (where affordant biting means biting the object which can be bitten, rather than anything at all). This behaviour happened after the robot had mastered looking at objects, and thus was able to look at the biteable object, and bite it. The same behaviour sequence appeared for bashing. Overall, this work demonstrates a quite general idea, which could be incorporated in other developmental systems. It would bring improvements to the speed at which developmental learners acquire behaviours, because, instead of simply taking many random actions, the selection of actions is guided to maximize the rate at which unknown regions are explored.

Bondu and Lemaire (2007) propose improvements to the Adaptive Curiosity mechanism of Oudeyer *et al.* (2007). Specifically, they propose a new way to select which zones the learner should focus on to improve its predictions. They draw parallels between Adaptive Curiosity and 'active learning'. Active learning is a type of supervised learning where, instead of the trainer feeding the examples into the system in some order, the learning program can ask for the examples which it believes will accelerate the progress of learning. Bondu and Lemaire (2007) propose criteria to balance exploration and exploitation, so that the agent does not expose itself to situations it does not learn from, and also does not waste time in situations which it already knows well. These new criteria are tested in a toy example as well as on real data sets for both diabetes tracking as well as credit approval; the results were better than those reported by active learning approaches from the literature.

Many researchers in developmental AI aim to build systems which would exhibit the emergence of stage transition. A different approach is proposed by Lee *et al.* (2007) where stages of behaviour are deliberately introduced as a way to facilitate ongoing development, by forcing the system to devote its attention to a simpler constrained learning problem at each stage. Lee *et al.* propose that a developmental system be designed with a schedule describing how constraints should be lifted in an ordered fashion as the system develops. They also cite significant evidence of constraints in human infancy, which are lifted as the infant develops; these constraints simplify the infant's learning problem at each stage.

Lee *et al.*'s experimental system has a robot arm to manipulate blocks, and a vision system. An example constraint is that the vision is disabled while the arm learns its initial movements. There is also a constraint on the sensory resolution, such that it is initially constrained to be coarse, and later becomes more refined. Lee *et al.*'s development approach follows a 'Lift-Constraint, Act, Saturate' cycle. When acting, the system explores the space of actions, given the current constraints, and eventually reaches a saturation where nothing more of interest (novel) remains to be explored. Then it is time to lift a constraint and the cycle begins again. For this reason, we group this work along with Oudeyer *et al.*'s; both are progressing to a new level because of satiety at the existing level, which can be viewed as an intrinsic motivation. Lee *et al.*'s system progressed from

basic groping behaviour (to explore its space), to eventually performing directed touching of objects in its environment, and cycles of sequences of touching.

Lee *et al.*'s approach to stage transition is different from most other works (which we discuss below in Section 4) because it requires that the stage transitions be 'engineered' via the schedule of constraint lifting. Most of the other works desire autonomous emergence of transitions, and have hopes of ongoing emergence of increasingly sophisticated behaviour. However, Lee *et al.*'s approach would seem to be an excellent approach to assisting the bootstrapping of a developmental AI system in the early stages.

3.4 Skills in reinforcement learning

As seen in Oudeyer *et al.*'s work above, it can be beneficial to have an agent devote its attention to learning skills in an initial phase, and then move on to more complex behaviours utilizing those skills. For example, first, the skills of learning to look, and learning to bite, and then later the more complex behaviour of coordinating biting with looking at the biteable object. This section will focus on the reinforcement learning approach to the learning of skills, which can then be used as subroutines to solve a larger learning problem. Skills can be captured nicely by the reinforcement learning technique of *options* Sutton *et al.* (1999). An option is a subroutine which can be treated in a similar way to a primitive action, in that from certain states an agent may have a choice between several primitive actions, or options. Once invoked, the option has its own policy to control the agent, until the option is finished, or interrupted.

The first work we review here is also aiming to model intrinsic motivation. Barto *et al.* (2004) note that most of the approaches to option learning merely extract options from the policy being learnt in an ongoing larger learning task (e.g. McGovern, 2002), rather than learning options for their own sake, or their own intrinsic interestingness. In Barto *et al.* (2004) and Singh *et al.* (2004), an agent must learn skills in a simple 5×5 grid world. There are individual skills which the agent must learn, and when the agent has mastered these, he can put them together in combinations to achieve more complex behaviours. The agent is given intrinsic motivation via certain *salient* events. A salient event includes changes in light and sound intensity. Each first encounter with a salient event causes a new option to be created to learn how to control that event. Intrinsic reward is provided each time the agent reduces its error in the prediction of a salient event. This means that the agent will try to improve a skill until it has mastered it and then will move on to another behaviour where its prediction error is larger. This is enough to motivate the agent to learn all the required skills, and eventually to put them together in more complex behaviours. Stout *et al.* (2005) use the same techniques in a larger gridworld, with similar positive results. These works on intrinsic motivation are similar to Oudeyer *et al.*'s in spirit if not in detail. Both are sufficiently generic to be combined with other developmental architectures; however, Oudeyer *et al.*'s is more autonomous in that it is building up skills in a bottom up fashion without any preconceived notion of what they should lead to.

Schembri *et al.* (2007a) propose an improvement on Barto *et al.*'s work to overcome the fact that salient events must be described by the programmer, rather than being identified by the agent itself. They propose that a 'reinforcer' maps experienced states to a level of saliency, where the reinforcer is a neural network whose weights are evolved by a genetic algorithm. In addition, they divide the agent's life into childhood and adulthood phases where childhood is spent learning basic skills with rewards given by intrinsic motivation, whereas adulthood is spent solving externally rewarded tasks which require combinations of the skills learnt in childhood. Schembri *et al.* (2007b) additionally look at allowing the length of childhood, and the parameters of learning, to be evolved.

Bakker and Schmidhuber (2004) present another work which attempts to learn options autonomously, without human guidance about what options will be useful later. They use clustering to create high-level abstractions of the state space; these then define regions of the state space, and it is the job of the low-level policies (i.e. options) to find how to move from one region

to another. The high level sets a new region as a goal, and a low-level policy will be learnt to get there. The example presented in Bakker and Schmidhuber (2004) is an office environment where an agent must navigate between rooms to find a goal location. The abstractions found by the clustering stage corresponded to the rooms and also certain corridor areas. The application of clustering is novel, and this approach would be worth investigating in a developmental learner which is building up skills hierarchically, as in the CLA Chaput (2004). We noted that Chaput did use the self-organizing map technique, but the clustering abilities were not exploited there (i.e. it did not consider regions of schemas which were close in the map).

Şimşek and Barto (2006) allow an agent to focus on the mastering of a skill itself, rather than learning it merely as a means to an end. They also propose a two-stage learning procedure which begins with a training phase in which skills are mastered before using that skill as a component in other more complex behaviours. They give an algorithm which can give an intrinsic reward so as to optimize exploration (rather than balancing exploration and exploitation). This is proposed as a method for focusing the agent's attention on completely learning a skill, before moving on to using that skill in goal-directed behaviour. Şimşek and Barto are particularly concerned with learning skills efficiently, and see the potential use of their algorithm as a component of an agent that autonomously builds a hierarchy of reusable skills.

Konidaris and Barto (2007) tackle the problem of learning options which will be 'transferable'; that is, which will be useful on other problems, where the state space is different. Their approach makes use of an 'agent space' as well as a 'problem space'; this is easy to illustrate using one of the examples Konidaris and Barto have used. The example problem consists of a sequence of rectangular rooms which the agent must navigate through, with locked doors, locks, and keys. Each of these items (doors, locks, and keys) emanates a specific-coloured light, and the agent has sensors on each of the four sides of its body to detect how close it is to a coloured light. These sensors form the agent-space description, and the agent can use this to learn options for getting the key and unlocking the door, regardless of which room it is in. The problem-space description has access to the actual x, y coordinates of the agent, and thus can learn how to navigate through the rooms. Obviously, the options learnt in agent space are equally useful in other problem spaces where there may be a different array of rooms. The use of coloured lights does perhaps oversimplify for the agent, but the idea of splitting off an 'agent space' is likely to be quite valuable in general. More generally, one could envisage a system which learns options relative to objects perceived in the world, so that these options would be useful in scenarios where the objects' positions are changed.

Konidaris and Barto (2008) note the difficulty faced by reinforcement learning if trying to learn skills in a high-dimensional state space. Progress is typically made by manually reducing the space to only a small number of relevant features; however, this is not possible if we desire truly autonomous skill acquisition. Konidaris and Barto (2008) tackle this problem by providing an algorithm which an agent itself can use to find an appropriate state space for learning a particular skill. It also allows the agent to switch between state spaces as appropriate to the skill being learnt. They define a set of abstractions of both the sensor space and the motor space. Their algorithm then finds an appropriate combination of sensor and motor abstractions which gives the lowest error for the skill being learnt. This allows the agent to select an appropriate state space for a skill, although the authors do note a shortcoming in that the possible sensor and motor abstractions must be provided; ideally, a completely autonomous skill learner might be expected to find its own abstractions.

Finally, there are also some other alternative approaches to hierarchical learning within the reinforcement learning literature. A survey of hierarchical reinforcement learning (HRL) is given by Barto and Mahadevan (2003). This describes two alternatives to 'options' for temporal abstraction and hierarchical control: the hierarchy of abstract machines (HAMs) and the MAXQ framework. Barto and Mahadevan (2003) also describe more recent work such as learning by maintaining a hierarchical memory of interactions. The memory allows agents to overcome the problem of hidden variables in partially observable worlds, whereas its hierarchical nature relieves the agent of the burden of maintaining long sequences of raw sensorimotor observations.

An example of this is given in Hernandez-Gardiol and Mahadevan (2000), which builds on the HAM approach and describes an HRL system which maintains a hierarchical memory of its interactions. The example in the paper is a hallway navigation robot which has two levels in its memory hierarchy: the lower level has procedures to navigate corridors, and the top level makes decisions at intersections.

Taking an overview of the approaches in this subsection, we see that the learning of hierarchies of increasingly complicated skills would seem to be exactly what is required for early developmental AI, and we see that reinforcement learning with options seems to be a fitting technique for this. Indeed, Sutton's PEAK project (Sutton, 2006a) aims to use options as well as PSRs to represent high-level human world knowledge in terms of (complex relationships among) low-level sensorimotor experience. The area of HRL has not yet reached this level of sophisticated knowledge being built from a complex hierarchy, going right down to sensorimotor primitives. Although one can imagine a hierarchy with many levels, in practice most reported examples have only two levels: one lower level on which basic skills are learnt and a higher level in which they are used to perform some more sophisticated behaviour. Ultimately, the approaches in this section are not complete solutions for developmental learning; they are useful techniques, but they need to be combined with some overall architecture which dictates how the different levels should be put together.

3.5 Bootstrapping without innate knowledge

Another approach to developmental AI (and the commonsense learning problem) is 'bootstrap learning'. This means giving the learner no innate knowledge, but instead providing an array of sophisticated learning techniques so that it can begin to find some order in the streams of data coming from its sensors. Having learnt some basic patterns, these serve as a grounding on which to begin learning higher level knowledge. This is in contrast to most of the works discussed earlier, where some basic knowledge would be provided in the form of innate schemas. In addition, the earlier works typically abstracted the world into discrete states and actions, making the learner's problem relatively easy, whereas the works reviewed in this section expose the learner to raw sensor data. Kuipers explains his rationale for pursuing the bootstrapping approach by explaining that even if we claim that some basic knowledge should be innate, we must then be able to explain how it was learnt by evolution.

> It is tempting to try to escape this problem by assuming that the foundational representations are innate when the individual is born, and so need not be learned. But this only pushes the learning problem onto the species, which must learn this knowledge over evolutionary time. We believe that in many ways developmental learning and evolutionary learning are similar, except that search is depth-first in one and breadth-first in the other. So we pretend that all learning is done by the individual, and postpone the decision of where to place the evolutionary/developmental boundary.[3]

3.5.1 Navigating in an unknown environment

Kuipers set out to make a learning agent which could be attached to any robot (with sensors and motors) situated in some environment, and which could learn how to sense the environment, move around, and discover some high-level knowledge about the layout of that environment. To solve this problem, Pierce and Kuipers (1997) developed a generic learning agent which uses a sequence of learning methods to learn a hierarchical model of a robot's sensorimotor apparatus and its environment. The hierarchy is part of the 'Spatial Semantic Hierarchy' (Kuipers *et al.*, 2000), which ultimately results in the learning of a cognitive map of the robot's world, although we will restrict our focus of attention to the lower levels of learning for our review here. The learning

[3] Introduction to bootstrap learning from `http://www.cs.utexas.edu/users/qr/robotics/bootstrap-learning.html`

works in three stages: first, the agent learns about its sensors; second, its motors; and finally, it learns behaviours which it can use to find places and paths in its world.

At the lowest level, the learner discovers *features* which are functions of the raw sensor data. An example is a group feature, which consists of a group of sensors that have similar values and frequency distributions. The learner then uses statistical methods to discover an approximation of the actual physical layout of its groups of sensors. Having the sensor knowledge as a base, the robot is ready to learn about its motor apparatus. It must learn the types of motion that can be produced and how to produce each type. A rough example of how this can proceed is if the robot generates a motion which causes it to pass an object; with its sensor knowledge, the robot can detect an edge of the object passing by distance sensors.

With this basic knowledge in place, the agent is ready to learn behaviours. Path-following behaviours are learnt by learning to control a feature. For example, if a feature corresponds to distance from a wall, then the agent merely needs to learn how to move while keeping this constant (wall following). In addition to path following, the agent also learns homing behaviours, to move to a position from where path following can begin.

Path following and homing behaviours can lead to a higher level knowledge of the world in terms of a finite set of 'actions' and 'views'. The 'actions' are simply the path-following and homing behaviours, and the 'views' are places in the world where these behaviours terminate, for example, if following a wall until a corner. This leads to a knowledge of schemas of the form view–action–view, which is similar to Drescher's context–action–result. The agent attempts to learn complete knowledge of the results of taking all actions from all views. Looking at the progress as a whole, it has successfully made the transition from a continuous world to a discrete world. This comprises the lowest levels of the Spatial Semantic Hierarchy; it later goes on to build topological maps.

Kuipers and Beeson (2002) build on this work to improve an agent's ability to distinguish places that look the same (but are different), and also to recognize places that are the same but look different. They introduce clustering algorithms for the initial recognition of places (i.e. to cluster similar sensory images), and then employ the Spatial Semantic Hierarchy to build the topological map, which allows similar looking (but different) places to be separated. With all the sensory images correctly assigned to places, the agent can then use a supervised learning algorithm to learn how to classify sensory images as belonging to the appropriate place.

In the SODA (self-organizing distinctive-state abstraction) method, Provost *et al.* (2006) use more generic learning methods to achieve the same goals as above, that is, moving from low-level sensory information to higher-level information about the robot's environment which it can use to navigate. The high-level perceptual features of the environment are learnt by using self-organizing maps. Hill-climbing and trajectory-following behaviours are then learnt, and grouped into high-level actions. The high-level actions are modelled using the reinforcement learning technique of options (see Section 3.4 above). Now that the agent can operate with these high-level actions and perceptions, it uses reinforcement learning to learn policies for navigating in its environment. This work is given a more rigorous grounding in hierarchical reinforcement learning in Provost *et al.* (2007).

Stronger and Stone (2006) give an alternative approach to learning to navigate in a world, when the robot's sensors and actions are unknown initially. They show that given an accurate knowledge of actions, sensor knowledge can be learnt by using polynomial regression to match the unknown sensor function. Likewise, they show that a model of robot actions can be learnt, if an accurate model of the sensors is known. They then show that both can be learnt simultaneously; although the initial learning of each is inaccurate, the models of sensors and actions get progressively more accurate as each helps to improve the other in a bootstrapping process. The system was evaluated with a Sony Aibo.

Olsson *et al.* (2006) revisit the learning of the sensor layout in Pierce and Kuipers (1997), and describe methods which can improve on it making it applicable to a wider variety of sensor apparatuses. They show how information theory can be used to define a metric for computing distances between pairs of sensors. They also describe new methods to discover and compute information flow in non-rectangular sensors and even sensors of different modalities. The framework was validated

with a Sony Aibo. The robot performs random movements, and maps out its visual sensors; then it computes the optical flow and learns to perform visually guided movement with its head.

Overall, these works address something that has been missing from the previously reviewed works, and that is the connection to low-level sensor data from a robot, without employing a human to perform the abstraction for the robot. These works clearly fit within the developmental framework as the low-level behaviours learnt can be plugged in as atomic entities in a higher level learning such as how to navigate an area.

3.5.2 Interacting with objects

Modayil and Kuipers (2004) tackle the issue of learning about objects in the world using the bootstrap approach. This learns 'shape models' for objects in its environment, from sensory snapshots of the object as it is tracked. Once the robot has a few shape models, it is able to categorize subsequent objects. This is extended in Modayil and Kuipers (2007) by learning percepts (such as angle and distance to object, and shape), and then forming concepts from percepts that are stable in time. This led to accurate classification of objects. The robot also learnt actions that could be applied to objects by initially performing motor babbling, and finding sequences which led to qualitative changes in a single percept. This allowed the robot to learn actions such as moving towards an object or pushing it. The objects were office furniture, such as a bin (pushable) or a chair (not pushable).

In Mugan and Kuipers (2008), an agent learns to interact with an object by finding significant 'landmarks' in its continuous input variables. It has been tested in a domain where a simulated robot arm tries to push a block in a specified direction. Its input variables include the position of the hand and the block, the position of the hand relative to the block, and the distance between them. The system learns triples similar to Drescher's context–action–result. Context elements can be continuous variables which are discretized into qualitative intervals such as negative, zero, or positive. With learning, the agent can find new landmarks in the continuous ranges. This is done by keeping a record of the activations of each rule, and then picking appropriate landmarks either to improve rule accuracy or based on statistics of values that correlate with events.

Having discretized the sensor space, reinforcement learning can be used to try to achieve a context. The state space for the reinforcement learning (RL) problem is given by all the possible discrete values for the input variables in the context. This work thus gives a solution to the problem of applying reinforcement learning to continuous domains, and the solution involves the agent discretizing the domain itself, in a way appropriate to the phenomena it is interested in. This is in contrast to the typical approach of relying on a human designer to find a suitable representation of the state space.

Stober and Kuipers (2008) describe an alternative approach to allowing an agent to autonomously discretize its inputs in order to be able to apply reinforcement learning. This work again used qualitative discretizations for the continuous variables, but also introduced methods for refining the representation. There were 10 variables, not all of which might be necessary for defining the state space. A 'leave one in' method was found to be successful, and worked by determining the policy contribution of each variable one by one, and then keeping the most useful ones.

As with the works on navigating in Section 3.5.1, these works on interacting with objects bridge a gap between symbolic developmental programs (such as in Section 3.1) and programs that deal with real sensor values. It remains to be seen how well the techniques can work together in a challenging environment.

3.5.3 Learning concepts from time-series data

Cohen *et al.* (1997) state in their earlier work that they differ from Brooks in that they do not feel that a physical embodiment, as opposed to simulation, is absolutely necessary; nevertheless, they also did some work with robots. These robot learning works can be classified as bootstrap learning, because they frame the problem in a similar way to Kuipers.

> Picture yourself in a room illuminated only by a computer screen, onto which bit strings flash, two or three a second, for a few seconds at a stretch. After a few moments of this, the screen goes

> blank, until it starts up again with another sequence of bit strings. Your task is to make sense of the bit strings. (Cohen *et al*., 2000)

Rosenstein and Cohen (1999) used clustering techniques to find concepts in the time-series data recorded from robot sensors. After a number of interactions with a range of objects, the system was able to cluster together the time-series segments which corresponded to interactions with particular objects, and to find prototypes for each category. The work effectively demonstrated the possibility of using unsupervised learning to enable a program to learn meaningful categories from uninterpreted sensor readings. Cohen *et al.* (2000) (see also Ramoni *et al*., 2002) again used unsupervised clustering techniques to find patterns in a mobile robot's input streams. They begin his time by learning Markov Chains to model the possible transitions between values for each sensor, and then apply clustering to find clusters corresponding to experiences of the robot (experiences include passing an object, or moving closer to an object). Firoiu and Cohen (1999) present an approach to finding logical representations for a robot's experiences, using the technique of HMM (hidden Markov model) learning.

One problem with clustering time series data from sensors is that two similar episodes could produce quite different time series. Consider a robot moving towards a wall slowly or quickly. The time series produced may be of different lengths and may appear dissimilar, but ideally we would like these two to be clustered together. Oates *et al*. (2000) tackle this by using dynamic time warping (DTW); this can stretch or compress parts of a sequence to find a match with a similar sequence. Efficient algorithms for DTW make it feasible for the authors to compute the pairwise matches for all possible pairs in a set of sequences. Clustering was applied after DTW. The clusters found accorded well with a set of human-made clusters, although there were some cases where DTW found similarities where humans would not. One drawback of time warping (noted in Cohen *et al*., 2002) is that it requires the user to first identify episode boundaries in time series.

Having learnt the clusters which correspond to distinct experiences, the above work was later extended (Schmill *et al*., 2000) to learn the initial conditions which are required for a particular action to lead to a particular experience. To learn these initial conditions, the sensor readings for one second leading up to each experience were recorded. This (together with the clusters) then formed the training data for a decision tree induction algorithm. The induced trees could not always predict the experience which would follow an action because of the partially observable nature of the problem. For example, the robot had no way of knowing, if moving towards a red object would lead to pushing it or being stopped by it. Nevertheless, the initial conditions were good enough to allow a simple planner to be implemented. King and Oates (2001) were also interested in planning, and to this end built on the DTW ideas; they also attempted to allow a robot to learn regions of the parameter spaces of its controllers which led to qualitatively distinct sensory outcomes.

Most of the works reviewed in the early part of this survey had very simplistic atomic actions which are complete in a single time step. If developmental learning is to be applied in domains where actions extend over time, and can be adjusted, and have varied effects, then it seems essential to have some methods to process the resulting time series. This concludes our review of bootstrap learning, and in summary we conclude that it fills an important gap in creating abstractions from low-level sensor data. We believe that a developmental learner will need to deal with low-level sensor data in complex domains, and learn abstractions for itself, if it is to be able to capture many commonsense concepts (arguments for this appear in the discussion in Section 4). Therefore, the earlier developmental learners could benefit by incorporating some of these bootstrapping techniques in their architectures. How well the combination will work together will need to be determined by experiment.

3.6 Learning affordances

Stoytchev (2007) describes a developmental approach to the problem of enabling a robot to learn about the affordances of tools. Stoytchev motivates this developmental approach by noting that

> … the developmental sequence leading to autonomous tool use is surprisingly uniform across different primate species. It follows the developmental sequence outlined by Piaget. Most of the

> observed variations between species deal with the duration of the individual stages and not with their order. This lack of variation suggests that evolution has stumbled upon a developmental solution to the problem of autonomous tool use which works for many different organisms. This increases the probability that the same sequence may work for robots as well. (references removed)

The following is the developmental sequence which can be used by autonomous robots to acquire tool-using abilities, according to Stoytchev: First, the robot learns the perceptual stimuli which are produced by its own body, and also learns to distinguish these from stimuli produced by other objects. Second, the robot learns the patterns of perceptions produced by the body, for example, patterns of movement; this can be used to control movement and predict it. Third, the robot learns about other objects in the world, for example, that they can be grasped and moved. Fourth, the robot learns how the actions performed with one object can affect another object; this leads to learning the affordances of tools.

Stoytchev's robot builds a model of its own body by doing random 'motor babbling', and observing the effects. Once it has a representation of its body, it can be extended when a tool is picked up. If a tool is used, the robot can morph its representation of its body so that it considers the tool as an extension of its hand. This is achieved by transforming markers in its body representation, so that they coincide with markers on the tool. The robot then learns tool affordances via exploratory behaviour, that is, using behaviours it has acquired in the pre-tool learning phases. The robot is presented with a tool on a table, and a hockey puck which can be moved with the tool. Five types of tool were used: a straight stick, an L-shaped one, an L-shape with an extra hook (as in a serif-font L), a T-shape, and a T-shape with extra hooks (as in a serif-font T). The robot can try exploratory behaviours (e.g. push, pull, or move to either side) and observe the results. A table describing affordances is maintained which can then be used to predict the effects of using various tools on the object. It can also be used to create behavioural sequences to achieve specific goals. In a recent work, Sinapov and Stoytchev (2008) build on these techniques to allow the robot to autonomously construct a taxonomy of the tools it experiments with, based on the effects those tools have on the puck being manipulated.

In his evaluation of the work, Stoytchev (2007) describes some shortcomings. First, the discovery of affordances is limited by the exploratory behaviours in the robot's repertoire; it will not find an affordance of a tool, if that requires it to perform a behaviour it lacks. To overcome this, Stoytchev proposes that the ability to imitate a human trainer could be incorporated, or the ability to learn new exploratory behaviours. Second, Stoytchev notes that the robot does not recognize the shapes of the tools; it can only discover their affordances by trying behaviours with them and observing the results. If a new tool is presented which is similar to an existing tool, one would like the robot to be able to make use of this and to experiment with behaviours suggested by the shape of the tool. However, the robot has no way of knowing what affordances it might have except by trying out its whole repertoire of behaviours.

The developmental approach to tool use could conceivably be added to many of the approaches we have seen above, as a later stage of learning, after basic behaviours with an agent's limbs have been learnt. We would require that the patterns of movement of the agent's own body (produced by these behaviours) be known by the agent. It would then be possible to extend the body representation when a tool is picked up. As for the shortcomings of the work, we have seen that there are developmental approaches which can learn behaviours without being pre-programmed (see, e.g. Section 3.5.2). This would address the second shortcoming Stoytchev noted, and potentially allow arbitrary new behaviours to be discovered with a tool.

4 Discussion

In this section, we revisit the reviewed works and assess the progress made thus far with respect to the ultimate goal of developmental AI. That goal is to build an AI system which could develop cognitively, and continue to develop, acquiring new knowledge through interactions with an

environment, and ultimately attaining a reasonable level of human commonsense knowledge. At present, this goal seems still a long way off, and therefore we attempt to pinpoint what is missing from existing work. We outline some key problems which have yet to be addressed, and whose solutions are essential to achieve the goals of the developmental approach.

We may assess the existing work by comparison with what has originally inspired the developmental approach, that is, the development of commonsense knowledge in an individual human. However, most of the AI work is not attempting to copy something that is exactly faithful to the path of human cognitive development, but rather to take inspiration from some of the abstract principles which it exhibits. We focus on two of these abstract principles: (i) emergent 'stage transitions' and (ii) learning 'physical world knowledge'. These are both essential for a successful developmental AI system. Emergent stage transition means that the system autonomously progresses to qualitatively different (and more sophisticated) types of behaviour; furthermore, this development must be ongoing. This is essential, as the developmental approach seeks to avoid attempting to handcode advanced behaviours and knowledge, and this means that the system must bootstrap itself through increasingly sophisticated stages of behaviour. Learning physical world knowledge is essential for the same reason: it is central to the developmental approach that this be learnt rather than handcoded.

4.1 Stage transition and ongoing emergence

In order to assess AI's progress in achieving stage transition, we first look at the main features of stage transition in human cognitive development to see if these features are present in the AI work. A detailed account of human stage transition appears in the work of Piaget (1936). A brief description of Piaget's sensorimotor stages is as follows. There are six sensorimotor stages, and we will overview the first five: (i) reflex actions, such as sucking what is in the mouth, or grasping what is in the hand. (ii) Acquired habits, such as sucking the thumb, looking at objects, and eventually grasping seen objects to take them to the mouth. (iii) Repeating interesting actions discovered by chance, for example, pulling a string to shake a rattle. Note that during this stage, a large repertoire of schemas[4] is acquired, for example, striking, scratching, pushing, pulling, etc. (iv) Means-end combinations of schemas, that is, the infant can combine schemas so that one schema is used as a 'means' and the other as an 'end'. A classic example of this is the behaviour of pulling aside an obstacle that blocks his arm in order to retrieve a toy that was visible, but behind the obstacle. (v) Experimentation to create new schemas, that is, the infant can vary the way a schema is performed, and this leads to the discovery of new schemas. For example, instead of simply hitting an object, the infant could learn to push it at one corner to make it tilt up.

Let us now look in detail at the stage transitions from 4 to 5 to see what features characterize stage transition. At sensorimotor stage 4, the infant has a collection of schemas which he can draw on to achieve goals. The stage 4 infant can make novel combinations among the schemas in his repertoire, but what he cannot do is invent a new means schema if a particular task requires this. In contrast, a stage 5 infant can experiment with his repertoire of schemas and come up with new ones for a particular task. An example of this is the task of using a stick as a tool to retrieve an object that is out of reach. If a stage 4 infant is shown this behaviour he may try to copy it, but will only be able to crudely hit the object with the stick, without adjusting the manner of hitting in order to make the object move closer. A stage 5 infant will also initially hit the object crudely, but will then be able to interpret the little movements of the object, and adjust the manner of hitting in order to make the object move closer. Piaget (1936) explains the onset of this experimental ability as being due to the infant having more schemas in his repertoire; these schemas are able to recognize the effects seen when an action leads to new effects, and thus the infant's attention is focused on finding the causes of these effects, and thus experimenting with the original action.

[4] Piaget's *schema* concept has been introduced in Section 3.1.

Looking at this type of stage transition, we can abstract two important features which characterize it. First, we see that it is not merely a case of learning a set of skills at one level and then moving to a higher level where they can be used as macro actions. Instead, it is a case of the existing skills taking on a new character, being used in a new way, leading them to spin out yet more new skills. A second feature that is evident is that the move to the new stage of behaviour happens because the level of development at the old stage enables the infant to notice some new relationship in the world. It is not the case that the agent itself deems (via some internal measure) that the competence at a particular skill has reached a certain threshold, and it is time to move to another level. Instead, the transition to a higher level is a function of the external world, and the interaction with it. Indeed, the higher level relationship is present in the external world already, and it is only when the agent's lower level competences reach a sufficient level that he can see this relationship (e.g. the relationship between hitting in a particular way and the resulting motion of the object). In Piaget's example of a stage 4 infant hitting the object with the stick, the infant is effectively blind to the displacement of the object; even if the object moves slightly closer, the infant is unable to interpret that motion, and is unaware that repeating it could bring the object into reach. Only when his schemas to recognize spatial movements develop further will he be able to profit from his observations in this scenario.

These two features of stage transition are also evident in Piaget's post infancy stage transitions, such as the move to concrete and then formal operations. We can also see analogous transitions in the development of human knowledge. For example, when early hominids got the idea of making tools which could shape other tools, this led to an explosion of new tools, where existing ones were modified with new techniques. Likewise, when the scientific method was adopted by modern humans, it led to existing ideas being adapted, and used in new ways.

Now, returning to the developmental AI literature, it is clear that this type of stage transition has not been captured. Many of the hierarchical reinforcement learning approaches rely on hierarchical levels which are decided in advance, and it is then a case of learning low-level skills at one level, and employing them at the higher level. This does not qualify as autonomous development. Oudeyer *et al.*'s (2007) work, however, does display the autonomous emergence of a higher level behaviour based on lower ones, when the Aibo coordinates biting and looking, to bite the biteable object while looking at it. However, it lacks the type of radical stage transition where existing skills could be used in a new way, leading to their own modification and the spinning out of new skills (thus, the first feature above is absent). The second feature is also absent because Oudeyer *et al.*'s (2007) system moves to a new stage of behaviour when it is satisfied that its behaviour on lower-level behaviours is good enough, and it is of little benefit to keep doing them. In contrast, Piaget's infants move to a new behaviour because what they master at one stage enables them to see something new in the world. Looking at some of the Piagetian schema approaches, we do see that this second feature is present. For example, Chaput's (2004) foraging robot can create low-level schemas predicting some result for some action, and can then begin to see relationships among these schemas. Before the low-level schemas are created, the robot is effectively blind to the relationships among them. However, the system does not seem to display the first feature of stage transition; there does not seem to be a radical change affecting the way the existing schemas can be used. In conclusion, there does not seem to be a developmental AI system which convincingly embodies all the hallmarks of Piagetian stage transition.

Prince *et al.* (2005) have come to somewhat similar (negative) conclusions in their review of 'ongoing emergence'. Ongoing emergence could be seen to be roughly equivalent to achieving stage transition, because if we build a system that achieves the right type of stage transition, then it should be capable of also exhibiting ongoing emergence. Prince *et al.* list six criteria for 'ongoing emergence'. We will focus attention on the second criterion, that is that after new skills are acquired, then 'These new skills are incorporated into the agent's existing skill repertoire and form the basis from which further development can proceed'. Of the six criteria, this is the one that Prince *et al.* identify as being absent in the works they review (they state that they have not found 'full evidence' for it). This criterion is implied by the first feature of stage transition which we have

identified above. Our first feature requires that any skills learnt could later be adapted and used in new ways when a stage transition happens, which presupposes that the acquired skills are incorporated and form the basis for ongoing development. Prince *et al*. give some examples of systems that develop from one stage to another and then stop; they come to the conclusion that some existing work displays emergence, but not ongoing emergence.

The desire to demonstrate 'ongoing emergence' leads us to consider the sensorimotor apparatuses and domains with which developmental AI programs operate. Chaput's foraging robot has a very impoverished sensory apparatus, where it has only five binary sensors; furthermore, its domain simply consists of identical balls on a flat surface, which it can pick up. It may be that it is impossible to achieve the type of stage transition and ongoing emergence we are looking for in such a domain. It is difficult to imagine what kind of advanced knowledge the system could progress to (if it is to go further than it has). This problem applies to many of the works reviewed. The types of development occurring in Piagetian stage transitions seem to require analogies among similar schemas, and such analogies seem to require a richer domain than that used by many of the works in developmental AI. Another important aspect of any domain chosen for developmental AI is its closeness to the human experience. If the domain is too far away from that in which humans develop, then it is less likely that a program developed there would be able to learn human commonsense knowledge. Thus, the idea of defining desirable features of stage transition and implementing them in a system is not sufficient; we must also require that the domain used should be somewhat close to that in which humans develop.

This argument can be reinforced by imagining a program that satisfies the abstract hallmarks of ongoing emergence (or stage transition). In looking at the six criteria of Prince *et al*. (2005), one could imagine a chess program which satisfies them; for example, a program which learns new advantageous positions and finds relationships among them, leading to the identification of further positions, etc. Similarly, one could view a mathematical discovery program such as Colton's (2002) as exhibiting stage transition and ongoing emergence (and if one feels that this program does not quite meet the criteria, one could probably imagine an improved version which would). However, none of these programs are likely to be able to learn human commonsense knowledge. Thus, for a developmental AI program to make a convincing claim about exhibiting stage transition (or ongoing emergence), we would like to see not only the hallmarks of stage transition, but also the acquisition of abstract knowledge about the physical world which we are familiar with. The ultimate proof that a learning algorithm is appropriate for the type of development we require is if it can learn the beginnings of commonsense knowledge.

A question arises as to whether or not a program which develops knowledge in an abstract domain (such as chess or mathematical discovery) would embody learning techniques which would be useful for learning human commonsense knowledge. This could only be answered definitively by experiment. However, there are some grounds for suspecting that the learning techniques used in these abstract domains are quite different from those used to learn basic commonsense knowledge; this is based on the observation that the type of logical thinking applied to learn in these domains does not come naturally to humans and requires significant effort.

4.2 Physical world knowledge

Building a system which can learn physical world knowledge is desirable not only for its own sake, but also because there is evidence that much of our higher level human reasoning rests on our models of aspects of the physical world. Lakoff and Johnson (1980) show that many concepts which humans express in their language are based on analogies with concepts from the physical world; for example, concepts like containment (inside/outside), size and strength (large and strong), breakability, moving objects (is ahead, is to follow), etc. It would seem that learning about these basic physical concepts will be essential for any agent that hopes to develop an understanding of human commonsense knowledge. To build a program to learn these physical concepts, one would probably need a continuous domain, rather than a gridworld like Drescher's. A continuous

domain would allow an agent to appreciate the various degrees and nuances of these concepts at a fine-grained level.

Taking the concept of containment as an example; it would not be sufficient to merely have a representation which has a binary notion of one object being contained or not contained in another. One would also need to understand ideas such as being partially contained, or something too big to fit, or more than one thing being squeezed into a container, or the idea of something leaking out of a container, etc. All of these are used in metaphors in explaining many other everyday human concepts. These ideas could be discovered by an agent if it had experience of the physical interactions involved; then the agent could build an internal representation of these interactions, and thus find analogies with other concepts. Furthermore, if an agent has these experiences and appropriate representations of interactions with physical objects, then the agent can always revisit these experiences in the mind's eye, and imagine variations and modifications to them, to extend its concepts. This could be necessary when reasoning by analogy, or to match a description being explained by another agent. This type of mental representation and reasoning would also be essential to allow an agent to perform advanced tasks with tools. All of this points to the need for a sufficiently rich physical world for interaction.

In this discussion, we are arriving at a weaker form of Brooks's thesis; Brooks claimed that it was essential to build AI for real robots in the real world (Brooks, 1991). The reason he gave is that any simulation of a simplified world will allow researchers to delude themselves that they are solving AI problems, when in fact they are abstracting away the really difficult parts of the problems by suitable simplifications to the world. Our weaker claim is that AI must be built in a world that has certain essential features of the real world. Although we see no problem with simulated worlds in principle (this is why our claim is weaker than Brooks's), we can see that if the simulated world is too simple, then it is unlikely that an agent learning in that world will be able to acquire many commonsense concepts.

As with many of the works on Piagetian schemas, much of the work on image schemas also suffers from having an insufficiently rich domain to allow some commonsense concepts to be represented. The primitive tokens in Neo's (Cohen *et al*., 1997) streams are already at a high level, such as a 'rattle-like shape'. Amant *et al*. (2006) do include a concept of containment, but the containment schema merely has a fixed set of slots to be filled in: 'Container, In-Out, Surface, Content, Full-Empty'. Thus, the types of nuances mentioned above could not be captured. Furthermore, the schema here is given rather than learnt, and therefore it is difficult to see how the agent could extend it. It would seem that any knowledge we handcode into a program in this way is not particularly useful because the program does not really understand it, and by this we mean that it is not built from any sensorimotor experience of the robot. However, if the robot has a memory of the physical interactions through which it learnt the concept, then it seems feasible, by re-imagining these, that the agent could extend the ideas. It seems that the advantages of discovering (or inventing) a new concept are not confined to artificial systems, as illustrated by the following popular quote from Piaget:

> Every time we teach a child something, we keep him from inventing it himself.

This is an idea also stated by Sutton (2006b) in his 'Verification Principle':

> An AI system can create and maintain knowledge only to the extent that it can verify that knowledge itself.

In support of this, Sutton also cites the fact that handcoded knowledge could not be revisited by the agent to be modified, if circumstances change (the agent does not know what that knowledge was originally based on anyway). This is illustrated clearly by Stoytchev (2007) where he states that the robot is limited by the exploratory behaviours in its repertoire which were handcoded; it cannot invent new behaviours if needed. However, if the behaviours in the repertoire were learnt by an exploratory process, then this could be revisited later if needed to adjust a behaviour in the repertoire. In support of his verification principle, Sutton (2006b) additionally

points out that if we rely on humans to verify the knowledge, then the system can never grow bigger than what the programmers can handle. One of the works which does convincingly learn a concept which it could verify itself is the work of Rosenstein *et al.* (1997); the agent here learns the concept itself through interactions, and has a mental representation of it. In order to extend this system to develop further physical world knowledge, it would be interesting to see how this concept could be used to reason by analogy with other concepts (this would require a richer world most likely).

4.2.1 Spatial knowledge

Kuipers *et al.* (2006) note that common sense 'is built on knowledge of a few foundational domains, such as space, time, objects, action, causality, and so on', and that 'spatial knowledge is arguably the most fundamental of these foundational domains'. This is also consistent with Piaget's theories. However, Piaget sees the acquisition of a knowledge of space as a process which goes hand in hand with the acquisition of a knowledge of objects. He also quotes Léon Brunschvicg who states that to conceive of space consists first of all in furnishing it (Piaget, 1937: 182). Piaget's description of the infant's construction of spatial knowledge entails knowledge of the effects of displacements of objects, and their relative positions before and after, and especially groups of displacements. For example, a child without objective spatial knowledge does not know that a path from A to B and then from B to C is equivalent to a path from A to C. In addition, a child without objective spatial knowledge will not expect that something placed behind an object from one side could be retrieved from the other side. This knowledge is learnt firstly in 'near space' (the zone of action of the child's hands), and then extended to far space, and later in the formal operations period it can be extended even to celestial bodies. This type of spatial knowledge is something more abstract than what the Spatial Semantic Hierarchy attempts to learn, and indeed it has not been attempted by any of the works reviewed here. The challenge here for AI researchers is to appreciate how much of what we take for granted as human adults was in fact learnt when we were infants, and should probably be learnt by a developmental AI system.

4.2.2 Object knowledge

Drescher (1991) attempted to model the acquisition of object permanence with his synthetic item. However, as noted above, the simple gridworld he used limits the scope for ongoing emergence in this system; it is therefore unlikely that this system could be extended to learn more advanced physical world knowledge, or nuances of concepts.

The work of Mugan and Kuipers (2008) showed how an agent can learn to push a block; this is of interest because it does work in a richer simulated world than Drescher's, as we have been advocating. It autonomously makes its own discretization of this continuous world by finding significant landmarks. However, the input to the agent does not include any visualization of the scene, as a vision system might observe; the inputs received are merely x, y locations of significant objects. In order to incorporate this work in a richer development system, it would be necessary to introduce a vision system, and some sort of 'image schema' to record the trajectory at which the block travels. With this, it would be possible to extend the program to find analogies among similar schemas. This would then open possibilities for further ongoing development and stage transitions. With regard to tool use, Stoytchev (2007) himself notes the lack of visual tool shape recognition as a shortcoming in his evaluation of his own work, as shape recognition would open the possibility for seeing analogies among similar shapes. All of this points to the need for a sufficiently rich sensory world for the learner.

To summarize the discussion on the acquisition of physical world knowledge, we note that there are no works addressing this adequately, and many are oversimplifying the world such that complex and subtle concepts cannot be learnt. Furthermore, our conclusions about stage transition suggest that it would be more profitable for developmental AI work to first focus on developing detailed physical world knowledge rather than attempting to build systems which display some of the abstract hallmarks of stage transition.

4.3 What path to follow?

The above discussion is rather negative about the progress thus far, because essential issues such as stage transition, and concepts of the physical world, do not appear to have been captured convincingly by any developmental AI system. We now take a look at what might be an appropriate direction for future research to attempt to capture the appropriate knowledge. We then discuss some of the techniques above which appear very promising, and some techniques which are absent and seem to be required.

We have seen that a number of systems have the ability to build on the knowledge they possess, in a hierarchical manner. However, this does not guarantee that the system will learn useful knowledge that humans can relate to. There are many ways to build hierarchical learners, and many domains in which they can be allowed to run. Some learners may find obscure abstract knowledge which is of no practical utility, whereas some may be limited by their domain so that there are no interesting higher-level concepts which can be learnt. For example, consider Chaput's (2004) foraging robot, or Chang *et al.*'s (2006) task of chasing for a cat; these can demonstrate hierarchical learning (i.e. learning which builds on what it knows), but only up to a point; it is not likely that these are suitable domains to demonstrate the ongoing emergence of higher level knowledge.

The safest bet then would seem to be to follow the knowledge that humans build, in the same order; at least we know that this is a developmental path that works. However, this is not at all easy as we do not know exactly what knowledge is innate in the human, nor what knowledge is present at each stage of development. Psychology can help here with carefully designed experiments, but the continuing controversy over the few results that have been found thus far suggests that progress here will be slow (Haith, 1998; Cohen and Cashon, 2003). The situation is somewhat similar to tracing the origins of advanced organs and organisms through evolution. In looking at the final product, we may be unable to guess how it evolved, but when we find intermediates in the fossil record, then the path of development becomes more clear. Human adult concepts seem astoundingly complex, but hopefully we can find the intermediates by probing the knowledge of infants and children with appropriate experiments. There is also a case for examining how other animals learn, since many of the lower order functions of human commonsense knowledge are present in these animals.

A particularly tricky issue is how closely we should copy human (or animal) development. One would like to avoid copying human development exactly, lest we have to design a lifelike infant robot that grows like an infant. Instead, we would like a loose copy which follows the essential milestones in development. It seems plausible that one should be able to build a simple simulated world which has a sufficient overlap with the real world to permit physical concepts of the type described by Lakoff and Johnson (1980) to be learnt. An agent growing up in such a world might not share the exact concepts of humans, but there should be sufficient overlap to permit a great deal of human commonsense knowledge to be learnt, and to permit interesting interactions with humans.

As for the techniques to be used, reinforcement learning (especially with options) has come up many times in the reviewed works, and it would seem to be an excellent starting point. RL is good for learning policies to achieve specific goals, but we have also seen that it is not difficult to combine RL with intrinsic motivation, rather than hard-codia a specific goal (see Sections 3.3 and 3.4). Furthermore, there is some consensus from the Piagetians that a schema takes the form of a triple (context, action, result); this can also be integrated with reinforcement learning (see, e.g. Chaput, 2004; or Witkowski, 1997).

A notable gap in the existing works is a lack of techniques for analogy finding and pattern matching. Perhaps the closest work is that of Oates *et al.* (2000) which finds similar sequences using dynamic time warping. Analogical reasoning is essential for the types of stage transition discussed in Section 4.1; when something happens as the result of an action, it is necessary to recognize which of the existing schemas it is analogous to. Analogical reasoning also seems to be essential for building models of the world, and indeed one could say that intelligence is all about building models of the world; this then allows accurate prediction of the effects of actions, and

allows an agent to reason about what manipulation of the world is needed to achieve a desired effect. The existing works have perhaps not yet found the need for analogical reasoning, as they have not yet got as far as building elaborate models of the world; existing models are restricted to some correlation among a limited number of sensorimotor schemas (this is the start of model building).

A final point which has been apparent when reviewing the various works is that comparison is very difficult because (with the exception of Chaput (2004)) all the works carry out their experiments in completely different worlds. If the community could agree on some standard benchmark simulation worlds, then it would be possible to compare all the techniques, and to clearly identify techniques which are superior. This would allow for more sharing of architectures, and component modules, rather than each research team designing and building their own systems from scratch. However, it may be that the field is not yet sufficiently advanced to make this feasible. For any candidate simulated world, a plausible objection could be made that the world is not sufficiently rich to allow ongoing emergence to be demonstrated. This argument will always be possible until ongoing emergence can be demonstrated convincingly.

5 Conclusion

This article has reviewed the area of 'early developmental AI', by which we mean AI systems which attempt to build their own knowledge and abilities autonomously (starting with little innate knowledge), and to develop continuously to reach increasingly higher levels of knowledge. We outlined the major areas of work, which include Piagetian schema learning, Lakoff and Johnson-inspired image schema learning, intrinsic motivation approaches, hierarchical reinforcement learning approaches, bootstrap learning, and tool use. It is not an exhaustive review, but we have aimed to include representative works from the dominant approaches to early developmental AI. We then outlined some significant issues which have yet to be addressed. These include the notion of stage transition (e.g. as it appears in Piaget's theory), and the acquisition of physical world knowledge (e.g. the concepts which Lakoff and Johnson refer to).

Our main conclusions were that, first, there is a need to work with reasonably realistic physical worlds which do not oversimplify the learner's experience, and second, the most promising approach to developmental AI might be to try to emulate the path along which humans acquire knowledge, copying the main milestones in world knowledge achieved by infants. This approach would benefit greatly from further psychological results. The approach seems to be a safe way to ensure that the knowledge developed will be on the 'right track' (towards human commonsense knowledge), and it is also attractive as it gives a number of reasonable intermediate targets to aim for which should be achievable from our current knowledge. If we accept the premises which motivate research in developmental AI, then these intermediates should be on the road to a solution to the commonsense knowledge problem. To conclude the review, we can say that developmental AI is still at its early stages, and much remains to be investigated. We have seen many techniques which could potentially be combined to complement each other's strengths, but this work remains to be attempted.

Acknowledgements

The author thank Jeffrey Donner for providing links to many papers.

References

Amant, R. S., Morrison, C. T., Chang, Y.-H., Cohen, P. R. & Beal, C. R. 2006. An image schema language. In *International Conference on Cognitive Modeling*, Trieste, Italy, 634–640.

Bakker, B. & Schmidhuber, J. 2004. Hierarchical reinforcement learning based on subgoal discovery and subpolicy specialization. In *Proceedings of the 8th Conference on Intelligent Autonomous Systems, IAS-8*, Amsterdam, The Netherlands, 438–445.

Barto, A. G. & Mahadevan, S. 2003. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems* **130**(4), 341–379.

Barto, A. G., Singh, S. & Chentanez, N. 2004. Intrinsically motivated learning of hierarchical collections of skills. In *Proceedings of International Conference on Developmental Learning (ICDL)*, Cambridge, MA, 112–119.

Bondu, A. & Lemaire, V. 2007. Active learning using adaptive curiosity. In *Proceedings of the Seventh International Conference on Epigenetic Robotics*, Piscataway, New Jersey.

Brooks, R. A. 1991. Intelligence without representation. *Artificial Intelligence* **47**, 139–159.

Carey, S. 1991. Knowledge acquisition: enrichment or conceptual change? In *The Epigenesis of Mind: Essays in Biology and Cognition*, Carey, S. & Gelman, R. (eds). Erlbaum, 257–291.

Carey, S. 1992. The origin and evolution of everyday concepts. In *Cognitive Models of Science (Minnesota Studies in the Philosophy of Science, Vol. XV)*, Giere, R. (ed.). University of Minnesota Press, 89–128.

Carey, S. 2004. Bootstrapping and the origins of concepts. *Daedalus, Journal of the American Academy of Arts & Sciences* **Winter**, 59–68.

Chang, Y.-H., Cohen, P. R., Morrison, C. T., Amant, R. S. & Beal, C. R. 2006. Piagetian adaptation meets image schemas: the jean system. In *SAB*, Nolfi, S., Baldassarre, G., Calabretta, R., Hallam, J. C. T., Marocco, D., Meyer, J.-A., Miglino, O. & Parisi, D. (eds), Lecture Notes in Computer Science **4095**, 369–380. Springer.

Chaput, H. H. 2004. *The Constructivist Learning Architecture: A Model of Cognitive Development for Robust Autonomous Robots*. PhD thesis, AI Laboratory, The University of Texas at Austin. Supervisors: Kuipers and Miikkulainen.

Cohen, L. B. 1998. An information-processing approach to infant perception and cognition. In *The Development of Sensory, Motor, and Cognitive Capacities in Early Infancy*, Simion, F. & Butterworth, G. (eds). Psychology Press, 277–300.

Cohen, L. B. & Cashon, C. H. 2003. Infant perception and cognition. In *Comprehensive Handbook of Psychology. Volume 6, Developmental Psychology. II. Infancy*, Lerner, R., Easter-brooks, A. & Mistry, J. (eds). Wiley and Sons, 65–89.

Cohen, P. R., Sutton, C. & Burns, B. 2002. Learning effects of robot actions using temporal associations. In *The 2nd International Conference on Development and Learning (ICDL'02)*, Cambridge, Massachusetts, 96–101.

Cohen, P. R., Atkin, M. S., Oates, T. & Beal, C. R. 1997. Neo: learning conceptual knowledge by sensorimotor interaction with an environment. In *Proceedings of the first International Conference on Autonomous Agents*, Marina del Rey, California, 170–177.

Cohen, P. R., Ramoni, M., Sebastiani, P. & Warwick, J. 2000. *Unsupervised Clustering of Robot Activities: A Bayesian Approach*. Technical Report 00-51. University of Massachusetts Computer Science Department.

Cohen, P. R., Chang, Y.-H., Morrison, C. T. & Beal, C. R. 2007. Learning and transferring action schemas. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, Hyderabad, India, 720–725.

Colton, S. 2002. *Automated Theory Formation in Pure Mathematics*. Springer-Verlag.

Drescher, G. L. 1991. *Made-up Minds, A Constructivist Approach to Artificial Intelligence*. MIT Press.

Firoiu, L. & Cohen, P. R. 1999. Abstracting from robot sensor data using hidden markov models. In *Proceedings of the Sixteenth International Conference on Machine Learning*, San Francisco, CA, USA, 106–114.

Haith, M. M. 1998. Who put the cog in infant cognition: is rich interpretation too costly? *Infant Behavior and Development* **21**, 167–179.

Hernandez-Gardiol, N. & Mahadevan, S. 2000. Hierarchical memory-based reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS)*, Denver Colorado, 1047–1053.

Holmes, M. & Isbell, C. 2005. Schema learning: experience-based construction of predictive action models. In *Advances in Neural Information Processing Systems (NIPS)*, Vancouver, B.C., **17**, 585–562.

Holmes, M. P. & Isbell, C. L., Jr. 2006. Looping suffix tree-based inference of partially observable hidden state. In *Proceedings of the 23rd International Conference on Machine Learning*, New York, NY, USA, 409–416.

King, G. & Oates, T. 2001. The importance of being discrete: learning classes of actions and outcomes through interaction. In *Canadian Conference on AI*, Stroulia, E. & Matwin, S. (eds), Lecture Notes in Computer Science **2056**, 236–245. Springer.

Konidaris, G. & Barto, A. 2007. Building portable options: skill transfer in reinforcement learning. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, Hyderabad, India, 895–900.

Konidaris, G. & Barto, A. 2008. Sensorimotor abstraction selection for efficient, autonomous robot skill acquisition. In *Proceedings of the 7th IEEE International Conference on Development and Learning (ICDL)*, Monterey, California, 151–156.

Kuipers, B. & Beeson, P. 2002. Bootstrap learning for place recognition. In *Eighteenth National Conference on Artificial Intelligence*, American Association for Artificial Intelligence: Menlo Park, CA, USA, 174–180.

Kuipers, B., Browning, R., Gribble, B., Hewett, M. & Remolina, E. 2000. The spatial semantic hierarchy. *Artificial Intelligence* **119**, 191–233.

Kuipers, B., Beeson, P., Modayil, J. & Provost, J. 2006. Bootstrap learning of foundational representations. *Connection Science* **18**(2), 145–158.

Kulakov, A. & Stojanov, G. 2002. Structures, inner values, hierarchies and stages: essentials for developmental robot architectures. In *Proceedings of the 2nd International Workshop on Epigenetic Robotics – Lund University Cognitive Studies 94*, Edinburgh, Scotland, 63–69.

Lakoff, G. & Johnson, M. 1980. *Metaphors We Live By*. University of Chicago Press.

Lee, M. H., Meng, Q. & Chao, F. 2007. Staged competence learning in developmental robotics. *Adaptive Behavior – Animals, Animats, Software Agents, Robots, Adaptive Systems* **15**(3), 241–255.

Lungarella, M., Metta, G., Pfeifer, R. & Sandini, G. 2003. Developmental robotics: a survey. *Connection Science* **15**(4), 151–190.

Mandler, J. M. 1992. How to build a baby: II. conceptual primitives. *Psychological Review* **99**(4), 587–604.

McGovern, A. 2002. *Autonomous Discovery of Temporal Abstractions from Interaction with an Environment*. PhD thesis, University of Massachusetts.

Modayil, J. & Kuipers, B. J. 2004. Bootstrap learning for object discovery. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, Proceedings* **1**, 742–747.

Modayil, J. & Kuipers, B. 2007. Autonomous development of a grounded object ontology by a learning robot. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, Vancouver, British Columbia, Canada, 1095–1101.

Morrison, C. T., Oates, T. & King, G. 2001. Grounding the unobservable in the observable: The role and representation of hidden state in concept formation and refinement. In *AAAI Spring Symposium on Learning Grounded Representations*, Stanford, California, 45–49.

Mugan, J. & Kuipers, B. 2008. Continuous-domain reinforcement learning using a learned qualitative state representation. In *22nd International Workshop on Qualitative Reasoning*, Boulder, Colorado.

Oates, T., Schmill, M. D. & Cohen, P. R. 2000. A method for clustering the experiences of a mobile robot that accords with human judgments. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, Austin, Texas, 846–851.

Olsson, L., Nehaniv, C. L. & Polani, D. 2006. From unknown sensors and actuators to actions grounded in sensorimotor perceptions. *Connection Science* **18**(2), 121–144.

Oudeyer, P.-Y., Kaplan, F. & Hafner, V. 2007. Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation* **11**(6), 265–286.

Perotto, F. S. & Álvares, L. O. 2006. Learning regularities with a constructivist agent. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, New York, NY, USA, 807–809.

Perotto, F., Buisson, J. & Alvares, L. 2007. Constructivist anticipatory learning mechanism (CALM): dealing with partially deterministic and partially observable environments. In *Proceedings of the Seventh International Conference on Epigenetic Robotics, Piscataway*, NJ, USA, 117–127.

Piaget, J. 1936. *The Origins of Intelligence in Children*. Routledge & Kegan Paul.

Piaget, J. 1937. *The Construction of Reality in the Child*. Routledge & Kegan Paul.

Piaget, J. 1945. *Play, Dreams and Imitation in Childhood*. Heinemann.

Piaget, J. 1971. *Biology and Knowledge*. Edinburgh University Press.

Pierce, D. & Kuipers, B. 1997. Map learning with uninterpreted sensors and effectors. *Artificial Intelligence* **92**, 169–229.

Prince, C., Helder, N. & Hollich, G. 2005. Ongoing emergence: a core concept in epigenetic robotics. In *Proceedings of EpiRob'05 – International Conference on Epigenetic Robotics*, Berthouze, L., Kaplan, F., Kozima, H., Yano, H., Konczak, J., Metta, G., Nadel, J., Sandini, G., Stojanov, G. & Balkenius, C. (eds). Lund University Cognitive Studies, 63–70.

Provost, J., Kuipers, B. J. & Miikkulainen, R. 2006. Developing navigation behavior through self-organizing distinctive-state abstraction. *Connection Science* **18**(2), 159–172.

Provost, J., Kuipers, B. J. & Miikkulainen, R. 2007. Self-organizing distinctive state abstraction using options. In *Proceedings of the Seventh International Conference on Epigenetic Robotics*, Piscataway, New Jersey.

Ramoni, M., Sebastiani, P. & Cohen, P. 2002. Bayesian clustering by dynamics. *Machine Learning* **47**(1), 91–121.

Rosenstein, M. T. & Cohen, P. R. 1998. Concepts from time series. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence AAAI/IAAI*, Madison, Wisconsin, 739–745.

Rosenstein, M. T. & Cohen, P. R. 1999. Continuous categories for a mobile robot. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence AAAI/IAAI*, Orlando, Florida, 634–640.

Rosenstein, M. T., Cohen, P. R., Schmill, M. D. & Atkin, M. S. 1997. Action representation, prediction and concepts. In *Working Notes of the AAAI Workshop on Robots, Softbots, Immobots: Theories of Action, Planning and Control*, Providence, Rhode, Island.

Schembri, M., Mirolli, M. & Baldassarre, G. 2007a. Evolution and learning in an intrinsically motivated reinforcement learning robot. In *Advances in Artificial Life, 9th European Conference, ECAL*, e Costa, F. A., Rocha, L. M., Costa, E., Harvey, I. & Coutinho, A., (eds), Lecture Notes in Computer Science **4648**, 294–303. Springer.

Schembri, M., Mirolli, M. & Baldassarre, G. 2007b. Evolving childhood's length and learning parameters in an intrinsically motivated reinforcement learning robot. In *Proceedings of the Seventh International Conference on Epigenetic Robotics*, Piscataway, New Jersey.

Schlesinger, M. & Parisi, D. 2001. The agent-based approach: a new direction for computational models of development. *Developmental Review* **21**, 121–146.

Schmill, M. D., Oates, T. & Cohen, P. R. 2000. Learning planning operators in real-world, partially observable environments. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling*, Breckenridge, Colorado, 246–253.

Shultz, T. 2003. *Computational Developmental Psychology*. MIT Press.

Şimşek, O. & Barto, A. G. 2006. An intrinsic reward mechanism for efficient exploration. In *Proceedings of the 23rd International Conference on Machine Learning*, New York, NY, USA, 833–840.

Sinapov, J. & Stoytchev, A. 2008. Detecting the functional similarities between tools using a hierarchical representation of outcomes. In *Proceedings of the 7th IEEE International Conference on Development and Learning (ICDL)*, Monterey, CA, 91–96.

Singh, S., Barto, A. & Chentanez, N. 2004. Intrinsically motivated reinforcement learning. In *18th Annual Conference on Neural Information Processing Systems (NIPS)*, Vancouver, B.C., Canada.

Stober, J. & Kuipers, B. 2008. From pixels to policies: a bootstrapping agent. In *7th IEEE International Conference on Development and Learning (ICDL-08)*, Monterey, California, 103–108.

Stojanov, G. 2001. Petitagé: a case study in developmental robotics. In *Proceedings of Epigenetic Robotics 1*, Balkenius, C., Zlatev, J., Kozima, H., Dautenhahn, K. & Breazeal, C. (eds). Lund University Cognitive Science.

Stojanov, G. & Kulakov, A. 2003. Interactivist approach to representation in epigenetic agents. In *Proceedings of the Third International Workshop on Epigenetic Robotics*, Prince, C. G., Berthouze, L., Kozima, H., Bullock, D., Stojanov, G. & Balkenius, C. (eds). Lund University Cognitive Studies, 123–130.

Stojanov, G., Bozinovski, S. & Trajkovski, G. 1997. Interactionist-expectative view on agency and learning. *Mathematics and Computers in Simulation* **44**(3), 295–310.

Stout, A., Konidaris, G. D. & Barto, A. G. 2005. Intrinsically motivated reinforcement learning: a promising framework for developmental robot learning. In *The AAAI Spring Symposium on Developmental Robotics*, Stanford, California.

Stoytchev, A. 2007. *Robot Tool Behavior: A Developmental Approach To Autonomous Tool Use*. PhD thesis, College of Computing, Georgia Institute of Technology.

Stracuzzi, D. J. 2005. *Scalable Knowledge Acquisition through Memory Organization*. Helsinki University of Technology.

Stracuzzi, D. J. & Könik, T. 2008. A statistical approach to incremental induction of first-order hierarchical knowledge bases. In *Proceedings of the 18th International Conference on Inductive Logic Programming*, Berlin, Heidelberg, 279–296.

Stronger, D. & Stone, P. 2006. Towards autonomous sensor and actuator model induction on a mobile robot. *Connection Science* **18**(2), 97–119.

Sutton, R. S. 2006a. *The Peak Project*. (unpublished document). `http://www.cs.ualberta.ca/~sutton/papers/peak-abs.pdf`

Sutton, R. S. 2006b. *Verification, the Key to AI*. (unpublished document). `http://www.cs.ualberta.ca/~sutton/IncIdeas/KeytoAI.html`

Sutton, R. S., Precup, D. & Singh, S. 1999. Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* **112**(1–2), 181–211.

Turing, A. M. 1950. Computing machinery and intelligence. *Mind* **59**, 433–460.

Witkowski, M. 1997. *Schemes for Learning and Behaviour: A New Expectancy Model*. PhD thesis, Department of Computer Science, Queen Mary Westfield College, University of London.

Zlatev, J. & Balkenius, C. 2001. Introduction: why 'epigenetic robotics'? In *Epigenetic Robotics 1*, Balkenius, C., Zlatev, J., Kozima, H., Dautenhahn, K. & Breazeal, C. (eds). Lund University Cognitive Science.