

Skill Acquisition through Program-Level Imitation in a Real-Time Domain

Mark A. Wood, *Student Member, IEEE*, and Joanna J. Bryson

Abstract—This paper presents an imitation learning system capable of learning tasks in a complex dynamic real-time environment. In this paper we argue that social learning should be thought of as a special case of general skill learning, and that the biases it presents to the skill learning problem radically simplify learning for species with sufficient innate predisposition to harness this power. We decompose skill learning into four sub-problems, then show how a modification of Roy’s CELL system [1] can address all these problems simultaneously. Our system is demonstrated working in the domain of a real-time VR game, *Unreal Tournament*.

Index Terms—Imitation, Social Learning, Memetics, Language.

I. INTRODUCTION

HUMAN-like intelligence requires an enormous amount of knowledge — solutions to the hard problems of survival and reproduction, which for our species have come to involve complex social and technological manipulations. Some of these solutions are passed to us genetically, and some are learned by an individual through trial and error. For humans, one key source of knowledge is culture [2]. By *culture* here we mean any knowledge an agent derives from conspecifics by non-genetic means.

Social learning *may* have evolved because it is safer than individual learning. However, given the vast complexity and rapid change we observe in human culture, we might imagine it is also fundamentally more *efficient*, at least for humans. In order for this strategy of knowledge (or rather behaviour) acquisition to be more efficient, the process of social acquisition for the average individual must be significantly less time consuming than trial-and-error learning.

In this paper we demonstrate how imitation learning can in fact provide the extra information a skill-learning algorithm needs to converge quickly. We focus primarily on what Byrne and Russon call *program-level imitation* [3], that is, the acquisition of complex skills through social learning, rather than the *action-level* of fine, precisely-timed motor control often emphasised in the robotics literature [4], [5]. In this paper we begin by describing the requirements of social and skill learning. We then present an imitation learning system, COIL, which we base on an existing real-time social learning system for robots, CELL [1]. We test COIL on two different imitation tasks in the domain of a real-time VR game, *Unreal Tournament*. We then analyse the complexity of the learning issues we encounter, and discuss implications for real, full-time social learning systems like people.

Manuscript received 1 December, 2005; revised 1 May, 2006. This work was supported by the EPSRC.

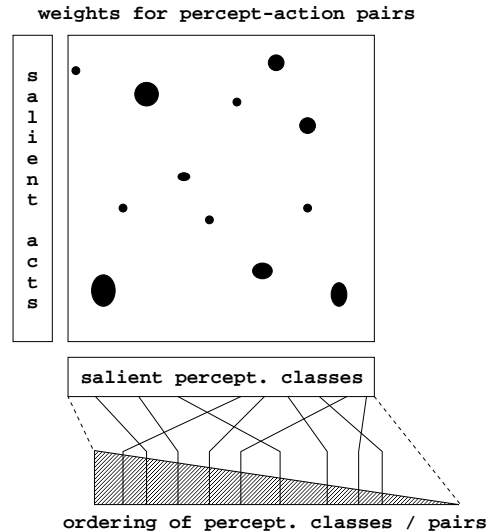


Fig. 1. Task learning requires learning four types of things: relevant categories of actions, relevant categories of perceptual contexts, associations between these, and a prioritised ordering of the pairings. Assuming there is no more than one action per perceptual class, ordering the perceptual classes is sufficient to order the pairs. See text for details.

II. TASK LEARNING

A. Requirements

There are at least four separate types of things that are learned in the process of learning a task (see Figure 1):

- 1) *Perceptual classes*: What contexts are relevant to selecting appropriate actions.
- 2) *Salient actions*: What sort of actions are likely to solve a problem.
- 3) *Perception-action pairings*: Which actions are appropriate in which salient contexts.
- 4) *Ordering of pairings*: It is possible that more than one salient perceptual class is present at the same time. In this case, an agent needs to know which one is most important to attend to in order to select the next appropriate action.

With respect to perception / action pairings, our research in primate task learning indicates that there should only be one action possible per salient perceptual context, but there may be many perceptual contexts in which a particular action may be relevant [6], [7]. Also note that although we mention perceptual contexts, we obviously do not mean the full context of all sensory information from a moment in time. Such a representation leads to a failure to generalise. Rather, detailed perception at any particular moment tends to be focused on a

few salient cues which will hopefully help disambiguate the current action-selection problem [8].

Clearly solving four problems simultaneously makes learning new skills a very hard problem, but equally it motivates social learning. In this paper we extend and analyse work showing that, in a social context, sensing and action categories can be recognised by their co-occurrence [1]. It should also be possible to induce sequential and hierarchical ordering [9], though we will not demonstrate that here.

B. Skill Acquisition via Imitation

We describe the process of imitation learning using two agents: the **expert** demonstrates the completion of some task, and the **imitator** learns while observing the expert acting, then attempts to act in kind. We use the term *expert*, as it implies that the imitator possesses skills worth acquiring, and that they will act broadly optimally (at least according to their own knowledge) when carrying out a task. We avoid the terms *model*, because of potential ambiguity with the underlying learning model, and *teacher*, because of the implication that the imitator is aware of and indeed actively assists the imitator [10], neither of which is necessarily the case.

Learning by imitation imposes several constraints which make the task-learning problem more tractable. For example, the imitator will observe a sequence of actions executed by the expert while carrying out the given task. These actions form a subspace of the imitator's whole action space, thus reducing the search time for salient actions. In contrast, individual learning strategies such as reinforcement learning [11] have by definition the disadvantage of the agent having to execute an action in order to discover its merit or, perhaps more crucially, its demerit. In our application domain of Unreal Tournament, this might amount to a bot having to kill a team-mate or fall into a ravine to receive the associated penalty and learn that this is a bad thing to do. Social learning therefore not only provides for faster and more practical learning, but also for *safer* learning [12], [13]. Individual learning algorithms also require the existence of a **reward function** [11]. These are not always easily constructed (see Section VIII-C below). Social learning of course on some level *requires* individual learning; here the reward function derives from replicating observed correlations.

This model of learning assumes of course that imitators are able to map the expert's actions into their own action space; this requires at least a partial solution to the *correspondence problem* [14]. Again, it is easier to see how this might have evolved in nature or be built in AI than many more complex reward functions. As we will see below, this does require a significant amount of endowed knowledge, though again there is considerable evidence that such endowment has biological equivalents [2], [15]. We discuss this further in Section VIII-B.

Since we assume that the expert is performing some (possibly dynamic and hierarchically structured) sequence of actions to the end of completing a task, we can also assume that certain actions are more likely to precede, co-occur with or follow certain environmental states. The imitator can use these observed temporal relations to calculate the most probable

causal links between perception and action. This satisfies the third sub-problem stated above. Note that finding suitable perception-action pairings both informs and is informed by the process of finding salient perceptual classes. This indicates that task learning is not a simple sequential process, but rather consists of an ongoing process of gathering data, and building and testing models. Such solutions are common now in machine learning [16], [17].

The final piece of the task learning puzzle, prioritisation, requires that the imitator recognise which action is executed when the environmental state simultaneously occupies multiple perceptual classes. While we don't demonstrate this here, we expect that a PCA-like analysis involving cross-referencing with previously learned (presumably more reliable) perception-action pairs will uncover lower-priority perception-action pairs. Here again co-learning priority should help facilitate and clarify otherwise noisy correlations for the other three classes of learning problem.

In theory then, imitation facilitates all four component sub-problems of task learning we described. Given some level of prior bias, the regularities expressed in the expert's performance may provide enough information to sufficiently constrain the learning problem to the extent that even with noise and ambiguity that result from being an external observer, the learner may learn faster with imitation than by trial and error alone. What remains is to develop a learning system specifically geared towards picking up on those regularities. To be plausible, this learning system should work in real-time on continuous-valued data for both sensing and acting. We now look at an existing language-learning robotic system which satisfies these desiderata, after which we will describe how it may be adapted for general-purpose task learning.

III. CELL: A WORKING LEARNING SYSTEM

The language-learning system in question is Deb Roy's CELL (**C**ross-channel **E**arly **L**exical **L**earning) system [1], [18]. As the name suggests, CELL was designed to emulate lexical acquisition in infants; specifically, to associate views of a given target object with spoken words describing that object.

Roy's experimental setup consisted of a camera mounted on an articulated robot arm which moved around an object to capture it visually from different angles. The sounds it was trained on were the utterances of a series of parents during sessions of natural¹ interaction with their infants, and their joint interaction with the target objects of the experiment. This occurred independently of the picture data capture, and the parents were not informed that their interactions should achieve any specific goals. A noise-cancelling microphone was used, and the utterances were subsequently digitally sampled for processing. The association of words to pictures was carried out artificially by an analyst noting the period during which a certain object was the focus of attention, and then pairing that period of sound with pictures of that object.

The CELL learning model consists of five main stages (see Figure 2 and details below in Section V):

¹At least, as natural as possible given the circumstances of a controlled laboratory experiment.

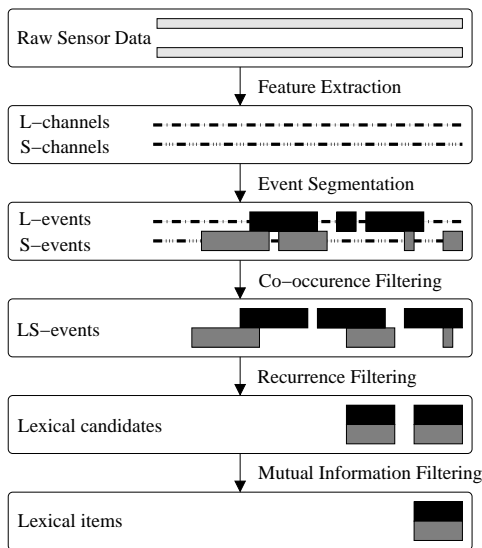


Fig. 2. The inputs and outputs of each stage of CELL (after Roy [1]).

- 1) *Feature Extraction* — salient features of the input sensor stream are extracted and isolated in separate Linguistic and Semantic data **channels**.
- 2) *Event Segmentation* — the channels are divided temporally into chunks called **events** and **subevents**.
- 3) *Co-occurrence Filtering* — Linguistic and Semantic chunks which co-occur are linked together and stored in a short-term memory buffer.
- 4) *Recurrence Filtering* — any paired chunks which are repeated in close temporal proximity are extracted and stored in a mid-term memory buffer. These pairs are **lexical candidates**.
- 5) *Mutual Information Filtering* — cross-channel Linguistic-Semantic mutual information is calculated for all lexical candidates. Those for which the value is sufficiently high are output (into a long-term memory buffer) as **lexical items**.

Comparing CELL with our task-learning model, we note that Roy’s system also completes, in a domain-restricted sense, most of the sub-tasks we described. If we assume CELL’s task is to assign spoken words to given visual input, then we can think of CELL’s action space as consisting of all possible vocalisations, and its perception space as containing all possible views (the converse is the case if we assume the task is to match objects to given speech). Candidate perceptual classes (visual representations of object properties) are created by applying the Feature Extraction and Event Segmentation stages of CELL to the camera data. A similar process applied to the microphone data discovers candidate actions (strings of phonemes). Co-occurrence, Recurrence and Mutual Information Filtering reduce these sets of candidates to those which are highly salient for completing the given task. Perception-action pairing also results from these stages of processing. Finally, a prioritisation for the pairs may be inferred from the mutual information value calculated during CELL’s final stage: the higher the mutual information, the higher the priority.

Looking at language learning in this way, one could argue that CELL is already capable of imitation learning, again in a domain-restricted sense. Knowledge about how words and objects are linked is copied from external demonstrations of these links by an expert. However, language learning is clearly a special case of social learning, in that it is *necessarily* social, because a learned lexicon has much less utility outside of the social context². There is no implication that the observed *behaviour* of the expert (ie. holding up an object and repeating its name) is particularly useful or should be generally adopted. Also, in some ways, CELL is not sufficiently general for task-learning. For example, language is by its nature sequential, but a task may require performing two actions simultaneously.

IV. LEARNING SCOPE IN DIFFERENT DOMAINS

These facts motivate our adaptation of CELL for application to more generic imitation tasks. To differentiate our adapted model from Roy’s original, we will refer to it as **COIL** — **C**ross-channel **O**bservation and **I**mitation **L**earning.

To explain this adaptation we need again to consider the more general definitions of perception and action, as described in Section II-A. To model human-like learning, as Roy did, we have chosen to impose similar constraints upon the system as would be present for human imitation. Specifically, we assume COIL is embedded in an embodied imitator agent which remotely observes an embodied expert agent acting in a shared environment. Bearing in mind the goal of task learning by imitation, data gathered by the imitator can be categorised into two channel types: Action and Perception. The Action channels receive data relating to the actions executed by the expert while completing the task. The Perception channels receive data relating to the perception of the expert, which is then used to determine the context of actions. Of course, the context determining actions includes internal state unlikely to be observable by the imitator, as well as external state which, while visible, will not perfectly correspond to the expert’s view.

As Roy points out [1, ch. 2], human infants possess innate biases which make learning tractable (see also [19]). This is reflected in CELL by, for example, the extraction of shape and colour characteristics from captured images, and the automatic recognition of phoneme boundaries. COIL is no different: the success of the Feature Extraction and Event Segmentation stages depend upon the imitator’s biases to filter out extraneous sensor information and parse continuous behavioural / perceptual data streams into representative categories [20]. Thus, the data going into the ‘main’ processing stages of COIL (Co-occurrence, Recurrence and Mutual Information Filtering) consist of select, segmented Action and Perception channels. Details of these stages can be found below, but the key question for now is: what is the output of the model?

In high-level terms, the resultant chunks stored in COIL’s long-term memory buffer represent actions paired with perceptions. Since the goal of an imitator is (presumably) to act, we could view these chunks as building blocks for a specification

²Though note again that the identification of salient categories implicit in label learning may be useful to an agent, even if that agent never speaks.

of imitated behaviour. If a given perception has been seen to instigate some action, this could be described as **motivation**. Conversely, if an executed action has been seen to bring about some perceptual state, this could be called **expectation**. Hence the output of COIL’s fifth stage consists of **Motivation Items** and **Expectation Items**, which I henceforth refer to collectively as **M-E Items**. These chunks are similar to Drescher’s *context/action/result* schemas [21], except that COIL does not go so far as to merge its observed *context/action* (Motivation) and *action/result* (Expectation) pairs. The issue of deciding how to act based upon M-E Items is beyond the scope of CELL’s original framework, but, since it is such a crucial part of the imitation process, we have added extra stages to COIL specifically for this purpose (see Section V-F).

Note that M-E Items correspond only to sense-action pairs as described in our original model. Providing that perceptual categorisation is strict enough so as to always create mutually-exclusive perceptual categorisation, then prioritisation of the pairs is unnecessary. If, however, in order to be generally powerful and reusable M-E Items *do* sometimes include non-exclusive perceptual state, then a prioritisation will have to be inferred. This again goes beyond the scope of CELL.

In the next sections, we realise this abstract description of COIL. We first introduce our chosen task domain and the initial test behaviours to be imitated therein. This is followed by a detailed breakdown of each stage of the model, supported by implemented examples, and contrasted with examples from Roy’s implementation.

A. The Real World, Robotics and Realistic Simulations

Much recent imitation research makes use of robots (or at least robot simulators) as the platform to test new models and algorithms [22]–[24]. Robots have many advantages as experimental platforms: they operate in the real world, in real time and face many of the same problems as human imitators while being able to exploit similar constraints (e.g. the physics of gravity, optics and impact). Robots like humans must deal with noisy and incomplete sensor information, imperfect motor control, and dynamic, unpredictable surroundings.

On the other hand, many practical issues associated with robots can inhibit research. Robots may severely exaggerate the effects of noise since it is difficult to tune them to the precision achievable by a human infant learning hand-eye coordination. Maintaining them takes considerable cost and expertise that is orthogonal to artificial intelligence, often requiring special technicians in the laboratory. As a result of these constraints, robot tasks seldom approach anything like the complexity of animal behaviour. Animal behaviour is characterised by multiple, dynamic and conflicting goals in environments populated with agonistic agents. It also is characterised by phased or cyclic activity operating on multiple concurrent time courses. Because robot behaviour is currently so severely constrained, concentrating all research on these platforms can result in overlooking combinatorial issues in learning and action selection that might be quite accessible in other platforms, such as artificial life [25], [26].

For this reason, we have opted for what we believe is a ‘best of both worlds’ domain: real-time 3D VR computer

games. This is certainly not a new approach, and there are many AI researchers already working in this domain [27]–[29]. Although games do avoid some of the technical problems of robots, they do introduce others. Again, perception and action may be far less reliable than for a skilled animal, and further they can be unreliable in ways that are bizarre by animal standards (e.g. failure to report the presence of a wall blocking a movement.) However, they are real-time and highly dynamic environments which require the pursuit of multiple goals (defeating aggressive opponents, curing injuries, accumulating weapons and/or other tokens, rescuing innocents, assisting teammates). They contain both continuous and discrete actions and perceptions. Also, importantly, they are not constructed by the experimenters as biased “toy” domains [30]. Rather, they present tasks on a general-purpose platform that are challenging for even human intelligence.

We now describe the game we have chosen, *Unreal Tournament*, highlighting its suitability for our purposes.

B. Unreal Tournament

Unreal Tournament (UT) is a commercially released, multi-player ‘First Person Shooter’ [31]. As the term suggests, the user has an agent’s-eye view of the game and direct, real-time control of an avatar’s actions. UT also supports remote control of agents by sending commands to the game server over a network. This provides a framework for allowing external programs to direct an agent’s actions. Such AI-controlled agents are commonly known as **bots** in the literature and gaming community. The game server, in turn, sends two categories of sensor data back to the client. The first is synchronous: at regular intervals the client is informed of the agent’s status (e.g. health, ammo, current weapon, etc). The second is asynchronous: for example whenever a wall is bumped, a footstep is heard or damage is taken.

UT offers the opportunity for real-time interaction and learning in a quasi-real-world environment. The game engine simulates physics, albeit simplified, such as collisions (of bounding polyhedra), gravity, and sound and light transmission. The bots’ sensor data are incomplete in the sense that only a reduced subset of the game variables are observable; the bots have limited virtual sensors. For example, the imitator cannot know the health state of the expert, although this may well affect the expert’s choices. However, what sensors are available are not subject to noise in the same way physical sensors would be. Neither are bots hampered by imperfect motor control. In fact, low-level movement (ie. that of arms or legs) is dealt with by the graphics engine, and bots can only be externally manipulated at a higher level (ie. move forward or rotate). This may seem unrealistic to those familiar with fine gesture simulation (e.g. [32], but note that there is a good deal of neurological data indicating that the ‘higher’ level brain functions we are presumably simulating (e.g. frontal lobe control of behaviour) can also operate at a similar fairly gross abstract level [33], [34]). Much of the natural imitation literature does *not* deal with precise replication of gesture [3] — or even using the same effector on an affordance. For example, Custance *et al.* describe agents that imitate a

demonstrator pulling a peg out with fingers by pulling it out with their teeth [35].

In summary, we believe that UT is realistic enough to allow the study of human-like imitation, but simple enough to enable us to get at core learning problems relatively quickly. To make explaining the mechanics of the COIL model clearer, we first take a moment to set out the initial task behaviours that we designed to test COIL’s aptitude for imitation in this domain. We will then use these example tasks in the description of the COIL algorithm.

C. The First Task Scenarios

UT ships with an environment editor, *UnrealEd* [36], which we used to create game worlds containing (and, in fact, defined by) the features necessary to complete our initial tasks. These simplified domains are much like the simplified domains used by Roy [1] and many other developmental-learning researchers. For **Task 1**, the expert demonstrator was a human-controlled bot, and the imitator was the COIL system ‘embedded’ in an AI-controlled bot. The imitator was programmed to follow the expert, remaining a fixed distance behind and to one side, so as much perceptual information as possible was shared. The only data made available to the imitator was that received from its own sensors. Latent variables within the expert (such as health) were invisible, just as they are to humans playing the game.

The environment itself was a single cuboid cavern aligned with the world axes³. Near each corner, equidistant from the nearest two walls, was placed a **health vial**. Such vials are visible both on screen and on sensors, provided they fall within a bot’s field of view (*view cone*). Once picked up they disappear, reappearing (*re-spawning*) after a short fixed time interval. The task demonstrated by the expert was simply to locate and collect vials as quickly as possible.

The expert, imitator and cavern remained unchanged for **Task 2**, but the health vials were replaced by a total of sixteen bots. These bots were recognisably from two different teams, half were to be considered ‘friends’, and half ‘enemies’. The task demonstrated was to locate and fire at enemy bots while avoiding friendly ones. A bot which is hit by weapon fire takes damage, affecting its internal health score, and is ‘killed’ when this score reaches zero, disappearing from sensors. The bots themselves were unarmed and posed no threat to either the expert or the imitator.

A full account of the scope of these tasks to test the breadth of COIL’s imitation abilities will be given in Section VI. First we describe the detail of COIL’s modelling procedures in the following section.

V. THE COIL SYSTEM

Our first theoretical COIL model, as detailed below, is as close an adaption as we found possible of CELL to a broader imitation learning context. We now look at each stage in depth: its input, its processes and its output. To aid understanding, we also explain how each stage was applied to Task 1, and compare it with Roy’s implementation.

³So the floor plane was horizontal and the walls were vertical.

A. Feature Extraction

The initial input into the first stage of the model is in the form of ‘raw data’ from the imitator bot’s sensors. Each sensor cycle provides data about objects of note in the environment, specifically the imitator bot’s own state, and the visible or audible state (such as location or actions) of other bots, items, weapons, navigation nodes, projectiles, etc. These data are extracted and / or merged into different **channels**. Conversely, a given channel should receive data pertaining to some specific feature of interest in the environment. COIL allows a channel to be one of two types: **Action** or **Perception**. Henceforth, we will use the term **channel set** to refer to the group containing all channels of a given type.

Our goal for Task 1 was to extract a minimal but sufficient set of features to allow learning of the task. There were only two types of action necessary to complete the task effectively: turning and moving⁴. We thus needed to build biases for detecting these kinds of behaviour into our system (see Section IV), and so created two Action channels: one for monitoring rotation in the expert, and one for monitoring motion. Using a piece of memory state in the imitator bot, we designed an algorithm to detect change in attitude of the expert and output the signal to the rotation channel. A similar algorithm detected change in position relative to attitude and fed that data into the motion channel.

Due to the simplicity of the environment, only one type of perceptual information is required for completion of Task 1. We know that the only visible items are the vials to be collected, and that there are no other obstacles, so tracking the bearing of the nearest item to the centre of view is sufficient. This single Perception channel receives data from a more complex algorithm in the imitator bot which uses the item and bot sensors to calculate the relative bearing of each visible item to the expert, and returns the least of these.

Roy’s experiment utilised two sensors and a total of three channels. Microphone data were fed into a single Linguistic channel ready for phoneme analysis. Camera data were fed into two Semantic channels: one transformed into colour histograms, and one into shape histograms. Using noisy physical as opposed to clean virtual sensors has disadvantages (see Section IV-A), but applying COIL to a real-world problem is nevertheless an interesting open research area. As with Roy’s implementation, after feature extraction the relevant data resided in three channels. The next stage is to separate the streams into discrete events which would eventually form the basis for the desired M-E Items.

B. Event Segmentation

Once the relevant data have been diverted into channels, two levels of event detection occur. Top-level **event** boundaries span the given channel set and are determined by a condition on *all* these channels *simultaneously*. The resulting chunks are known as **A-events** or **P-events** (depending upon the channel set in question). Lower-level **segment** boundaries also span the channel set but, in contrast, are determined by conditions

⁴By *moving* we mean translational movement along the floor plane.

on *each* channel *individually*. In other words, every channel contributes segment boundaries that then span every other channel in the same channel set. **A-subevents** are consequently defined as any continuous sequence of segments within an A-event, across any subset of Action channels. **P-subevents** are analogously defined.

The Feature Extraction process assigned representative ‘labels’ (real numbers) to pre-defined classes within each channel. For the rotation channel, the classes were *turning anticlockwise* (represented by -1), *not rotating* (0) and *turning clockwise* (1). For the motion channel, they were *moving other than forward* (-1), *not moving* (0) and *moving forward* (1). The bearing channel classes were *no items visible* (100), *item anticlockwise* (-1), *item ahead* (0) and *item clockwise* (1). In addition to the choice of channels, the classes themselves also reflect the imitator’s innate knowledge brought to the task at hand and provide a bias toward tractable learning.

Given the data classes, designing event recognisers was relatively straightforward. **A-events** (events which span the Action channels) were triggered by any change in expert bot state. In other words, an A-event started whenever the expert moved or turned (or both) and ceased when it stopped. The gaps between A-events represented times in which the expert was still. A-event segment boundaries were triggered by a change in state in either of the Action channels, for example: a change in the direction of motion from forward to strafing sideways. **P-events** (events which span the Perception channels) were triggered by any change in the bearing channel state. So if, for example, an item that was previously clockwise of the expert becomes ahead, a new P-event commenced. In our implementation, P-events formed a continuous sequence, with the start of a new event triggering the end of the prior. No further segmentation of P-events occurred.

Linguistic-event (L-event) boundaries in Roy’s model were triggered by the commencement or cessation of speech input, and so the events themselves consisted of spoken utterances delimited by silence. Segment boundaries coincided with probable phoneme boundaries generated by a Recurrent Neural Network. Semantic-events (S-events) worked slightly differently in that they were not temporal, but a collection of static pictures taken of the object in question from different angles, or **object view sets**. As such, they had no constituent segments, and only contained subevents differentiated by channel span, ie. colour and shape. This allowed Roy to control the complexity contribution from at least one channel, which we could not.

C. Co-occurrence Filtering

Co-occurrence filtering is a simple procedure which searches the segmented channel sets for A-events and P-events which overlap in time. Such a pair of co-occurring events is termed an **AP-event**, and shunted to **Short Term Memory** (STM), which is implemented as a queue.

All the channels receive data concurrently due to the imitator’s fixed sensor cycle frequency⁵. Since P-events covered

the entire timeline for the duration of each simulation, every A-event overlapped with at least one P-event, although the inverse was not necessarily the case. When a coincident pair was found, they were copied to STM as an **AP-event**: a period of continuous action coupled with a period of uniform perception.

Roy’s object view sets were timestamped, as were the utterances, which allowed overlap to be established. Consequently, an LS-event in his model consisted of a spoken utterance paired with an object view set.

D. Recurrence Filtering

The arrival of an AP-event in STM initiates a comparison to be made between the new member and each of the existing members. The constituent A-events of the pair of AP-events under comparison have already been subdivided into segments. Using some predefined metric on A-subevents, d_a , every A-subevent from the new AP-event is compared with every A-subevent from the other. If the distance between them falls below a predetermined threshold, t_a , then the two subevents are marked as matching. The same process is carried out for P-subevents.

Given the matched pairs of subevents generated by the above process, the new AP-event is scanned for co-occurring matches. If a matched A-subevent coincides with a matched P-subevent, their partners in the other AP-event are checked for co-occurrence. If they too coincide, then a recurrent match has been found. Such a match is used to create an **M-E Candidate**, and these are stored in another buffer: **Mid Term Memory** (MTM).

An A-subevent in this instance is a continuous sequence of motion and / or rotation segments. To measure the distance between such segments, we defined the action distance metric initially as follows:

$$d_a(x, y) = \sum_{c \in C} \frac{|\bar{y}_c - \bar{x}_c|}{|C|} \quad (1)$$

where x and y are A-subevents in A-space, C is the set of all channels spanned by x and y , and \bar{x}_c and \bar{y}_c are the mean class values for the given channel c for each respective subevent.

For example, let both x and y be A-subevents representing uniform clockwise turns. So, $C = \{R\}$ (just the rotation channel), $|C| = 1$, $\bar{x}_R = 1$ and $\bar{y}_R = 1$. Then we have:

$$d_a(x, y) = \frac{|1 - 1|}{1} = 0 \quad (2)$$

in other words, the two A-subevents are coincident in A-space. If, however, we let y represent a uniform anti-clockwise turn (so $\bar{y}_R = -1$), we have:

$$d_a(x, y) = \frac{|1 - (-1)|}{1} = 2 \quad (3)$$

P-subevents are blocks of uniform perception, and the perception distance metric was defined similarly:

$$d_p(x, y) = |\bar{y}_B - \bar{x}_B| \quad (4)$$

⁵Again, this may not be as biologically implausible as it sounds; the brain seems to also work to synchronise sensory input [37].

No sum is necessary, since all P-subevents span and only span the bearing channel (B). The thresholds t_a and t_p were initially both set at 0, so only coincident A- and P-subevents were regarded as matching. Recall that AP-events in STM undergo recurrence filtering whenever a new AP-event arrives. When a pair of matching A-subevents co-occurs with a pair of matching P-subevents, then one of the A-subevents is taken as representative and coupled with one of the P-subevents. This couple is an **M-E Candidate**, which is then added to MTM. Examples of M-E Candidates could include a ‘clockwise turn’ coupled with an ‘object clockwise’, and a ‘forward motion’ coupled with an ‘object ahead’.

Bearing in mind that L-subevents in Roy’s implementation consisted of sequences of phonemes, the metric d_l that he used was an acoustic distance metric based on the likelihood that two sequences were generated by the same Hidden Markov Model. He restricted the search by only comparing phoneme sequences of less than one second duration containing at least one vowel. S-subevents were either colour or shape view sets in the form of histograms. The visual distance metric used, d_s , was based upon a χ^2 -test for histogram similarity. The match thresholds were set relatively low, and so many Lexical Candidates were created. The metrics Roy selected were well-established; unsurprisingly, there are no conventions for measuring the distance between two generic actions or perceptions. Also, Roy assumed ‘close’ temporal proximity of similar LS-events (justified by his study of highly repetitious infant-directed speech), and implemented STM as a queue of around just five items. We cannot make this assumption, as action-perception pairs which arise from a task demonstration could be separated by significant time intervals. Thus our STM was much bigger (typically 25 items), and this affected efficiency (see also Section VII).

E. Mutual Information Filtering

Providing the number of Lexical Candidates in MTM exceeds some fixed minimum, mutual information filtering occurs whenever recurrence filtering generates a new candidate. Before attempting to explain the process, some more terms need to be defined:

- **A-space** and **P-space** are metric spaces with metrics d_a and d_p respectively.
- An M-E Candidate is equivalent to a point in A-space, coupled with a point in P-space. These points are known as the candidate’s **A-** and **P-prototypes**.
- An **A-unit** is a sphere in A-space of radius r_a , with an A-prototype at its centre (**P-categories** are defined analogously).
- An A-unit coupled with a P-category is called an **M-E Item**.
- An M-E Candidate **matches** an M-E Item if the candidate’s A-prototype falls within the item’s A-unit, and the candidate’s P-prototype falls within the item’s P-category.

And so, the algorithm runs as follows:

- 1) An M-E Candidate is selected. Its A-unit and P-category are initialised to have sufficiently small radii so as to contain no other A- or P-prototypes.
- 2) r_a is increased until another A-prototype falls within the A-unit.
- 3) The mutual information for this configuration of radii is calculated [1, chapter 3]. If it exceeds the previous maximum, then it is stored along with the current radii configuration.
- 4) Steps 2 and 3 are repeated until every other A-prototype has been included in the A-unit.
- 5) r_p is increased until another P-prototype falls within the P-category. r_a is reset to its initial state.
- 6) Steps 2 to 5 are repeated until every other P-prototype has been included in the P-category.
- 7) Steps 1 to 6 are repeated for every M-E Candidate in MTM.
- 8) If the maximum mutual information exceeds some predefined threshold, then the M-E Candidate and its optimal radii are used to create an M-E Item (see above). This is stored in **Long Term Memory** (LTM). The M-E Candidate, and all those that match the new M-E Unit, are removed from MTM.

For Roy, an L-prototype was a phoneme sequence in L-space, the space of all such sequences. An L-unit was therefore a sphere of phoneme sequences that ‘sound sufficiently like’ the prototype. An S-prototype was a colour or shape view sets in S-space, the space of all such view sets. An S-category was thus a sphere of view sets that ‘look sufficiently like’ the prototype. So, a Lexical Unit was effectively a spoken word paired with a viewed object, both with allowances for individual variation. Roy used a linearly-interpolated prior to smooth mutual information values for infrequently observed pairs. This has not yet been implemented in COIL; in fact we are currently considering alternative methods of calculating mutual information (see Section VII-B).

F. Generating Behaviour

Half of the challenge of imitation is acting upon what has been learned, but this function is not incorporated into CELL (although Roy does practically demonstrate his acquired language knowledge in other applications [1, chapter 6]). Therefore we have added modules to COIL to complete the ‘imitation loop’ — to facilitate acting upon acquired behavioural knowledge. These modules, along with their theoretical basis, are the subject of this section.

We have seen how, through observing the completion of a given task by an expert, COIL can construct a ‘dictionary’ describing the apparent action-perception relationships that are required for, or are a consequence of, the completion of that task. However, the fundamental question of how to act upon said knowledge remains, and is nontrivial. To answer this question, we need to consider what the resultant M-E Items represent in more detail.

In section IV, we suggested that *motivation* implies a perception that triggers an action, and *expectation* implies an action that predicts a perception. Recall that an M-E Item is an A-unit coupled with a P-category, and that these, in turn, are derived from an A-subevent which coincides with a P-subevent. The priority of these subevents indicate whether a

given M-E Item is in fact a Motivation or Expectation Item. If the initiation of the corresponding P-subevent *preceded* the A-subevent (the perception preceded the action), then it is a Motivation Item; if the A-subevent was initiated first, then it is an Expectation Item.

We discuss to what extent COIL solves a *correspondence problem* [14] in Section VIII-B, but for now, suffice it to say that the expert and the imitator have *similar embodiment*; their avatars have identical body configurations in UT. This being the case, we can assume that an imitator capable of discerning the perceptual state of an expert, can use the same representation for its own perceptual state. This allows the imitator to search its acquired Motivation Items for those which match its state (via the P-category), and retrieve candidate actions for execution (via the A-unit). If the Motivation Items cover the imitator’s perception space, then the above method provides a complete specification for behaviour, albeit with a simple reactive mapping between state and action. Expectation Items are not as obviously applicable in terms of determining action, and for Tasks 1 and 2 the reactive behaviour provided by Motivation Items alone has been adequate for effective imitation. We are currently exploring the potential for using Expectation Items to test the correctness of learned behaviour, and for more complex planning in tandem with Motivation Items (see Section IX), but for now we look at our basic action selection method and its utility in completing Task 1.

Recall that the possible perceptual classes for Task 1 related to the possible positions of the nearest item to the centre of the bot’s view cone: *no items visible*, *item anticlockwise*, *item ahead* and *item clockwise*. Having observed the expert collect vials for some period of time, the imitator switches⁶ into acting mode and begins to ‘observe’ its own perceptual class instead of that of the expert. At the instance at which acting commences, the imitator searches LTM for Motivation Items which match its current state. If there is more than one, the Motivation Item with the highest mutual information value attached to it is chosen. If none match, then the imitator has not learned what to do in this circumstance; depending on the experimenter’s preference, it either returns to observing, or is coerced into taking a random action in the hope of entering a new perceptual class.

Having decided on a Motivation Item to ‘act upon’, the action that is ‘most representative’ of its component A-prototype is selected for execution. This is calculated by measuring the distance between the A-prototype and contrived ‘pure actions’ (that is, a set of A-subevents, $Y = \{y_i\}$, for which each \bar{y}_i is equal to an action class label — see Equation 1). The action classes which are comparable depend upon which channels are spanned by the A-prototype. The action closest to the A-prototype in A-space (measured using the action distance metric in Equation 1) is then executed.

For example, suppose that the selected Motivation Item contains an A-prototype recorded from the Rotation channel only, where the expert turned clockwise for 90% of the time, then paused for 5%, and finally turned anti-clockwise for 5%. The mean rotation class value for this A-prototype is:

$$\bar{x} = (0.05 \times -1) + (0.05 \times 0) + (0.9 \times 1) = 0.85 \quad (5)$$

bearing in mind that the rotation class labels (and thus the ‘pure actions’) are *turning anticlockwise* (−1), *not rotating* (0) and *turning clockwise* (1). So, the closest action is *turning clockwise*, which has a distance of:

$$d_a(0.85, 1) = |1 - 0.85| = 0.15 \quad (6)$$

The action selected for execution is therefore *turn clockwise*. The particular action classes that the imitator can recognise in the expert represent a bias in COIL for recognising actions in the imitator’s own repertoire. This can again be partly justified due to the similar embodiment of the imitator to the expert (see Section VIII-B). The command *turn clockwise* (with a parameter angle), for example, exists as a method in the bot controller module.

For Task 1, the imitator was programmed to retrieve a new action every time its perceptual state changed (checked at each sensor cycle), or to repeat the previous action if the state remained unchanged. We defined the actions typically to be short and discrete (such as *turn clockwise 20°*), for several reasons:

- 1) The sensor cycle rate was slow enough that at maximum rotation velocity (for example), whole perceptual classes could be turned through in between cycles, missing an opportunity to change course of action.
- 2) Most of the available bot commands must be given a numerical parameter, so true continuity of action is difficult.
- 3) The game server requires some delay between commands. For example, trying to execute an action every sensor cycle is impossible.

It is quite possible to build apparent continuous motion out of small discrete ‘decisions’, particularly for embodied agents where their physical plant does a certain amount of its own integration, though it is also possible to build a system with dedicated integration modules [38]–[41].

Clearly, the quality of the imitated behaviour depends upon both the biases we have built into COIL for the task in question, and the quality of the demonstration given. The next section looks at our results in this area, the problems we have overcome, and those we are yet to.

VI. PRELIMINARY RESULTS

Prior to giving our experimental results, it would seem appropriate to explain our choice of tasks in terms of hypothesising COIL’s ability to solve a wider range of imitation problems.

Task 1 is designed to be an elementary task to test the correct functioning of the different system components. Perception space is partitioned, so there are no concurrent competing perceptual classes. However, the observed actions (ie. turning and rotating) are real valued with varying duration and may overlay multiple perceptual classes: they are not naturally discrete. Also, the bots sense with respect to absolute (world)

⁶Done manually, but could be made automatic - see Section IX.

TABLE I
CORRECT BEHAVIOURS FOR TASK 1

Policy	Perceptual Class			
	Item ↘	Item ↑	Item ↗	Item ×
O1	Turn ↘	Move ↑	Turn ↗	Turn ↗
O2	Turn ↘	Move ↑	Turn ↗	Turn ↘
NO1	Turn ↗	Move ↑	Turn ↗	Turn ↗
NO2	Turn ↘	Move ↑	Turn ↘	Turn ↘

co-ordinates; COIL must translate these readings into expert-centric co-ordinates to allow easier recognition of the expert’s objectives. Task 2 requires all of the above, but additionally COIL must arbitrate attention by prioritising multiply satisfied perceptual classes (see Section VI-B for examples). Task 2 necessitates firing a weapon which, although being more easily discretisable, introduces a new problem: such a short action could get lost or ignored amongst long continuous ones, especially when the sensor sampling only occurs at around 10Hz. We believe that most imitation problems could be constructed by arbitrarily complex combinations of these problems; the question of how well COIL will scale remains to be investigated (see Section IX). We now recap the specific requirements of each task in turn, and give the results of COIL’s learning efforts.

A. Task 1

At the highest level, the behaviour to be imitated in Task 1 was to collect health vials. To achieve this, the imitator monitored the rotation and motion of the expert, and the relative position of the vials in the environment. There were thus two Action channels (*rotation* and *motion*) and one Perception channel (*bearing*). Broadly speaking, actions were delimited by a change of direction (or cessation of motion), and perception by a change of perceptual class. A full specification of Task 1 can be found in the examples of Section V.

As previously mentioned, the Perception space of Task 1 is divided into four mutually exclusive classes: *no items visible*, *item anticlockwise*, *item ahead* and *item clockwise*. COIL ideally must acquire sufficient Motivation Items, by observing the expert, to ‘cover’ this space with the correct actions to complete the task. The actions recognised by COIL were: *turning clockwise*, *turning anticlockwise*, *not turning*, *moving forward*, *moving other than forward* and *not moving*. Including the possibility of a null assignment arising from a gap in the observed behaviour, there are therefore seven possible assignments to each perceptual class, giving 28 possible pairings and 16384 possible behaviour **policies**, where a policy is a set of pairs covering all salient perceptual categories⁷. Of these, only four policies will correctly complete the task (see Table I).

Policies **O1** and **O2** are ‘optimal’ inasmuch as an agent acting accordingly will complete the task in minimal time

⁷*Moving other than forward* does not translate directly into an executable action. In our experiments, it was mapped to *move backward*, but never appeared in any of the learned behaviour. Also, although *not moving*, *not turning* and the null assignment all have the same practical outcome for the imitator, as far as COIL is concerned, they are different actions, which is why we make the distinction above.

TABLE II
RESULTS FOR TASK 1

Tactic	% correct behaviour				
	100	75	50	25	0
CW	5	3	2	0	0
ACW	4	6	0	0	0
Mix	5	5	0	0	0
Total	14	14	2	0	0
Mean correct behaviour: 85%					

and with minimal wasted ‘energy’. The ‘non-optimal’ policies **NO1** and **NO2** arise from the fact that turning clockwise x° is equivalent to turning anticlockwise $(360 - x)^\circ$. An agent adopting either of these policies will be able to complete the task, but, unless the imitator and vials have a very specific initial configuration, time and energy is likely to be wasted through the extra rotation. Any other policy will result in an inability to complete the task.

An experimental trial consisted of COIL observing, via an imitator bot, the demonstration of the task by a human-controlled bot for 60 seconds. As operators of the expert, we used three different ‘tactics’ to complete the task:

CW Tend to turn clockwise if no vials are visible.

ACW Tend to turn anticlockwise.

Mix No fixed tendency.

We carried out ten trials for each tactic, for a total of 30 trials. To discover the imitator’s learned behaviour, we queried the model directly with the four perceptual classes, rather than by observing the imitator act. Attempting the latter could have lead to further error and subjectivity affecting the results.

A ‘percentage correct behaviour’ score was assigned to each trial, based on comparisons of the learned policy with each of the four correct policies, taking the greatest number of matched perception-action pairs. Each policy comprises of four pairs, so the possible scores are 0%, 25%, 50%, 75% and 100%. Any behaviour which does not score 100% (ie. which mismatches all of the correct behaviours) will fail in the imitation task, but we have made no further attempt to assess how ‘badly’ one behaviour performs compared to another, except by these percentage scores. We felt further analysis would cloud the results we have and introduce a further unnecessary layer of human interpretation. The results are shown in Table II.

Note that nearly half of the 30 imitators learned a fully correct behaviour. The majority of the remainder formed only one incorrect pair; two formed two incorrect pairs (our worst case). Additionally, 27 of the 30 learned policies matched most closely to one of the optimal policies (15 matched O1, and 11 O2), with only 3 which more closely matched non-optimal policies (2 matched NO1, and 1 NO2). Clearly, COIL performs significantly better than random action allocation would, but what about individual learning methods, such as Reinforcement Learning? This question is addressed in Section VIII-C; for now we move onto the second task.

B. Task 2

Until now, Task 2 has only been described in brief (see Section IV-C), so the specification that follows adds enough detail to gauge COIL’s success, but omits the lowest level technicalities. For this task, the expert aimed to seek and destroy ‘enemy’ bots, while avoiding ‘friendly’ bots. The expert was armed, but the target bots were not. This time, the environment was encoded by two Perception channels: *bearing* and *affiliation*. The bearing channel functioned analogously to that in Task 1, replacing items with potential target bots. The affiliation channel received a classification of the nearest bot as either *friend*, *enemy* or *no target visible*. The Action channels available to the imitator were *rotation* (identical to Task 1) and *firing*. The firing channel received a binary signal, *firing* or *not firing*. As per Task 1, A-events were delimited by absences of action, and A-subevents by a class change on either A-channel. P-events were delimited by changes in the affiliation of the nearest target bot, and further segmented into P-subevents by changes of bearing class. The A-space and P-space metrics were defined very similarly to Task 1, with the notable exception of the firing channel: we defined the distance between *firing* and *not firing* to be significantly farther than other distances, to reduce the probability that the imitator would confuse the two states.

One of the key differences between the encoding of this task and the previous one is that there are now two Perception channels concurrently receiving data. Therefore, there are three perceptual classes ($[bearing = B]$, $[affiliation = A]$ and $[bearing = B \wedge affiliation = A]$) concurrently applicable to the expert, as opposed to one ($[bearing = B]$). COIL must learn to correctly prioritise these classes. For example, the imitator might observe the expert firing at an enemy:

$$[bearing = ahead \wedge affiliation = enemy] \Rightarrow firing$$

COIL could, however, assign *firing* to $[affiliation = enemy]$, which would result in the imitator opening fire before the target is in position, or to $[bearing = ahead]$, which would disastrously result in the imitator firing indiscriminately, at friends and enemies alike.

There are (at least) two ways of defining correct behaviour for this task. The first concentrates on the key states:

- KS1** $[bearing = ahead \wedge affiliation = enemy]$
KS2 $[bearing = ahead \wedge affiliation = friend]$

These are key, because correct behaviour in these states is most critical for completing the task. It is worth noting that, although we use the terms *enemy* and *friend*, to the imitator they have no intrinsic meaning; they are just two different types of target. We specify the four possible outcomes relating to these states, in descending order of merit, as follows:

- | | |
|----------------------|-----------------------------------|
| Good | Fire at enemies, avoid friends. |
| Safe | Avoid both enemies and friends. |
| Trigger-happy | Fire at both enemies and friends. |
| Bad | Fire at friends, avoid enemies. |

The second way of defining correct behaviour is attempting to define an optimal policy and then calculating percentage correct, as in Task 1. An optimal policy for this task should

TABLE III
RESULTS FOR TASK 2

Tactic	Key state behaviours				Mean % correct
	G	S	T	B	
CW	6	4	0	0	72
ACW	7.5	2	0.5	0	71
Mix	8	1	1	0	65
Total	21.5	7	1.5	0	69

be **Good** in the key states, turn toward a visible enemy, and turn (in either direction) when faced with a friend.

Each trial lasted as long as it took for the expert to eliminate all the enemy bots, typically approximately 60 seconds. Tactics **CW**, **ACW** and **Mix** were used analogously to Task 1, again with ten trials each for a total of 30. Results are shown in Table III. Decimals arise from the fact that, for this task, maximal mutual information was often shared by a number of Motivation Items. Where the number of Motivation Items for two actions were equal, the action assignment for that perceptual class was divided in two (clearly in practise, a method of selecting between these actions would need to be found). Note that over two thirds of the bots tested performed correctly in the key states, and the remainder acquired a definite tendency against shooting friends. The mean behaviour correct score was less for the Mix tactic, because the inconsistency in turning direction provided fewer similar examples for the imitators to form a fully correct policy.

The clear differentiation between firing at friends and firing at enemies shows that COIL has succeeded in prioritising concurrently applicable perceptual classes. Combining this result with that of Task 1, we have some foundation for constructing more complex behaviours to solve hierarchically-structured and multi-part tasks. On the other hand, the lack of perfect performance on such seemingly simple problems, and the fact that the algorithm as it stands will not *improve* performance if it gets off to the wrong start in learning are both troubling. We now consider the extent to which COIL (as an extension of CELL) is currently a plausible model for human-like imitation.

VII. ANALYSIS

Although COIL successfully learned some basic skills in UT, a number of problems arose from our attempt to be as faithful to the CELL template as possible. This in turn allows us to analyse our proposal of CELL as a model of social learning, and discuss how combinatorial issues are likely to affect the component algorithms as the task scales. In this section we identify the problems and propose solutions.

A. Representational Differences

COIL’s primary weakness has resulted from trying to ‘fit’ general representations of action and perception into spaces designed for speech and vision.

CELL receives continuous microphone data, later converted into discrete phonemes during Event Segmentation. COIL receives continuous action data which is parsed into discrete

action segments. The crucial difference is between *the spaces in which these discrete objects lie*. We omit the details of Roy’s metric here, but intuitively phonemes can have infinite variation and can be mapped into a continuous space where the notion of ‘nearness’ is well-defined. In contrast, the limited set of actions that can be initiated by a UT bot lie in a discrete space where ‘nearness’ is not only hard to define but not a particularly useful concept. For example, how far is **jump** from **fire**, and what use would that information be anyway?

Comparing Roy’s Semantic and our Perceptual Categories, another issue comes to light: segmentation of perception. Roy deals with this by providing CELL with static set of images (see Section V-B), but there is no equivalent get-around in COIL. Our system effectively forces the programmer to segment perception space in advance through the hard-coding of event and subevent triggers. These comparisons point toward using a more discrete, symbolic representation for both actions and perceptions *throughout* COIL. This argument is further strengthened by recent results we have obtained using a symbolic classifier; specifically a decision tree (DT). The classifier took a purely symbolic form of the M-E Candidates that arrived in MTM as input, and output an action map as described in Section V-F. Run on the same data that was recorded during the experiments above, the DT scored a perfect 100% in Task 1, with non-optimal behaviour learned in only one trial. Interestingly in Task 2, the DT always learned **Good** behaviour in the key states, but made almost as many mistakes on average as COIL (overall mean % correct behaviour 70%).

In summary, we believe that COIL must be tailored to deal with new discrete representations in order to make further progress, also potentially reducing the complexity of the matching and clustering algorithms inherited from CELL which are designed to function in continuous metric spaces.

B. Scope Differences

CELL is designed to emulate early lexical learning, so the task environment is expected to be constrained in certain important ways. One constraint that cannot easily be mapped into the broader domain in which COIL is expected to operate is the **Recurrence Filtering** constraint (see also Section V-D). This limits CELL’s ‘attention span’ to about five consecutive LS-events, and only word-concept pairs which recur within this frame survive the filter. It is, of course, perfectly reasonable to assume high-frequency repetition of keywords, given that the recorded speech is infant-directed. However, our UT imitators cannot always assume equivalent high-frequency repetition of action-perception pairs during the completion of a task: that quite depends upon the task. This begs the question of whether learning UT entirely socially would require ‘infant-directed violence’. Notably, some species of predator provide their young with extra practice for the final (and thus, least frequently occurring) stage of a hunt [10].

In our first implementation of COIL, we simply slackened the constraint by extending the attention frame to cover a ‘large’ number of AP-events. In fact, we initially removed it all together, allowing the filter to compare each new AP-event

with *all* of those that had previously been added to STM, but this proved too computationally inefficient. It may make more sense to remove the filter altogether and replace it with something more appropriate both to the learning problem and to the new representations mentioned above. One possibility is for short term memory to become some form of **episodic memory** [42]–[44]. For example, rather than storing the past five events, we might store an entire task-learning episode. Each game could consist of a sequence of episodes where different tasks are learned and perhaps returned to later. M-E Candidates created from events stored in episodic memory could then be stored in MTM for the duration of a game (or possibly a fixed number of games), and LTM would hold M-E Items generated during all the games of an agent’s lifetime.

Alternatively, episodic memory could be a new structure designed specifically to deal with discrete representations and *replace* both STM and MTM. An episode could be represented as a list of observed co-occurring A-subevents and P-subevents, with a count of the number of times that pair recurred with the episode. That way, each AP-event could be processed individually (ie. no pairwise comparisons would be needed), potentially reducing the computational complexity of the system as a whole. This is actually a reasonable match to indexical theories of the hippocampus’ role in memory [42], [44], [45]. These are best known for capacity reasons: a sparsely encoded representation allows the retention of many events in a finite neural memory. However, they provide the extra (and perhaps more important) attribute of generalised storage of commonly occurring events, which may thus accumulate more ‘weight’ in the representation (see Figure 1). This sort of representation might also solve the problem of assigning ‘nearness’ values statistically [46].

None of the above critique is aimed at discrediting Roy’s application of CELL *to the learning problem for which it was designed*. Rather, we simply call into question whether program-level imitation is one of the ‘variety of domains’ [1, p. 47] in which CELL is readily applicable. Roy makes no such direct claim, of course, and it may be that with the alterations given above and the extensions described in the next section COIL can become a robust, working system, but that remains to be seen.

VIII. DISCUSSION

A. Scalability

So far we have only applied COIL to two relatively simple ‘local’ tasks carried out independently of each other. Even these have highlighted some serious computational issues for imitation learning, which we analysed in the previous section. However, confronted with the ‘full’ world of UT, we think it unlikely that even an *ideal* version of COIL would succeed in learning a correct behaviour across a range of tasks. This is mainly due to the complexity of the perception space that would make this possible. We’ve already established that the number of executable actions is not a major issue, but in a flat COIL architecture, every goal and ‘thing worthy of attention’ (including memory) in every conceivable in-game task would necessitate its own perception channel.

The solution to this problem is almost certainly some system of different task-learning frameworks or spaces. To again reference the biological solution, we know that in rats the semantic referent of hippocampal ‘place’ cells are dependent on the task the subject believes it is engaged in [47]. Further, the developers of the two dominant cognitive modelling tools, Soar and ACT-R, have found that creating modular ‘work spaces’ is necessary for replicating human-like learning [48], [49]. As we discussed in Section II-B, we have some ideas about how to extend COIL to filter for the effects of prioritisation between co-occurring perceptual categories. However, this sort of mechanism would be limited in its ability to combat combinatorics, so we will probably also need a mechanism for creating new task contexts and swapping attention to them for both learning and acting. Again, in real life for a complex task, new recruits are generally trained on one aspect of a job at a time.

B. The Correspondence Problem

We now return to the matter (first mentioned in Section II-B) of the the correspondence problem [14]. The most obvious version of the correspondence problem is trivial for ‘Virtual Imitation’ — one bot imitates another identical bot, the mapping should be one-to-one, and the discussion is over. However, as our discussion of perception in the previous section indicates, the correspondence problem goes far deeper than simple body-part to body-part mappings. Taking the whole system into account, the agents *are* quite different, particularly in the areas of perception and intelligent control. The human’s actions (which guide the expert) are forcibly mapped via the bot controller to an avatar, while COIL’s action repertoire is restricted by and to the methods available in the bot controller. It is at this early stage that the bulk of action correspondence takes place, resulting in a near one-to-one relationship in the bots, although there are some actions which a human-controlled bot can take which are impossible for an AI-controlled bot, such as following a curved path.

In perception, although the human has a broadly first-person perspective (ie. from inside the expert bot’s head), the perception itself is not the same: the human makes decisions based on visual cues displayed on the screen, and only perceives relatively broad classes (such as *to the left* or *through that door*). COIL has access only to its bot’s virtual sensors which, although they attempt to give similar information to that available to a human player via the screen, use fundamentally different representations (ie. waypoints instead of walls / doors, precise co-ordinates for location, etc.).

The point at which these perceptual correspondences are smoothed over is also different. In designing the action channels and their classes, and then in building the behaviour generator, we largely solve the action correspondence problem for the agent, and like many researchers in AI we must be deeply grateful to those in Natural Intelligence who are finding justifications for the belief that these basic capacities do seem to exist, at least in primates [15], [50]. But for perception, the COIL system discovers its own correspondence. Indeed, it may well find regularities that are not exactly the ones the

human expert saw, but which serve as adequate indicators to shape a reasonable policy [51].

C. Comparison to Individual Learning

We now revisit the question raised at the end of Section VI-A as to how COIL compares with individual trial-and-error learning. Learning an optimal four-state, seven-action policy (such as that required for Task 1) is not a difficult task for most Reinforcement Learning algorithms, but it presumes the existence of a **reward function** [11]. In the absence of such a function, an RL algorithm cannot converge upon a successful policy, because it has no way of ascertaining which actions are good or bad. It would be impossible for RL to learn Task 1 without attaching a reward to collecting a health vial (or a penalty for not collecting one), and thus adding prior task knowledge to the agent⁸. The only true prior knowledge built into COIL is that the expert is to be imitated, no matter how intrinsically ‘rewarding’ that may or may not be. This all assumes the environment is fully observable, and if this is not the case then we venture into the realm of POMDPs [52], and the complexity rises by an order of magnitude. Conversely, we have shown COIL to operate in real-time in a partially observable⁹ environment.

A promising research approach more similar to our own is the work of a few researchers to add learning into complex planning frameworks *at the level of the planning*. This has been done on teleo-reactive plans in a flight simulator [53] and on STRIPS-like plans in a ‘physics correct’ blocks-world simulation [26]. Like our system, these require the specification of a great deal of information about the action and perceptual primitives in advance, but can then adjust this information to learn a task. A full analysis of the power and complexity of these various representations remains to be done.

D. COIL for Robots

Another potentially interesting study would be to return the COIL algorithm back to a robotic system, or at least a more realistic / robotic simulation. Besides providing a system of potential utility to real robotics, this would open the way to examining the interaction between program-level and action-level imitation. One problem that COIL as it stands is completely incapable of solving is when two non-discrete actions (that is, two actions requiring some duration such as turning or moving forwards) must occur in a sequence *with no change of perceptual context*. While these sorts of events may be rare in nature, it may also be that a solution to this problem would be to learn the whole gesture as a single action, triggered by the initial perceptual context. Although there is evidence that many primates are capable of programme-level imitation [3], [54] apparently only humans are capable of the sort of precise temporal representation found in action-level

⁸It could be argued that the biases built into COIL which afford a similar representation of the task environment to the expert constitute prior task knowledge. If this is the case, then that knowledge is at best rather weak and implicit, unlike the explicit reward function of an RL algorithm.

⁹Recall that much of the UT game state is latent - see Section IV-B.

imitation [55]. Perhaps the fact that we uniquely combine these two capacities explains why ours is the only species with rapidly accumulating culture.

IX. CONCLUSION

We have discussed why imitation matters — because social learning is central to the intelligence of many species, and because doing it quickly is critical in explaining the elaborate behaviour humans express. We have related a language learning model, CELL [1] to social learning in general and presented a new learning system, COIL, that extends it to the general problem of imitation learning. We have tested and demonstrated this system on a real-time agent learning to cope with a highly dynamic environment.

We are currently working on improving the efficiency and accuracy of cross-channel information calculation. There is also potential for using Expectation Items (thus far discarded) for both testing the correctness of learned behaviour and longer-term planning, through their predictive property. Enabling the imitator to automatically switch between observing and acting modes (based upon some confidence level) should yield interesting data regarding how long certain parts of a task take to learn, and which take multiple attempts to perfect. We may also add an individual-learning module to COIL, to study the interplay between individual and social learning. Scaling issues could be eased by introducing independent sets of hierarchical perceptual classes, allowing global and local goals to be dealt with separately.

Ultimately, our wish is to expose COIL bots to the rich problem space of a full UT game world. It is our belief that exploring the systemic and learning requirements for such agents will give us major insights into the uses and requirements of social learning.

ACKNOWLEDGEMENT

The authors would like to thank Will Lowe and Jan Dru-gowitsch for their clear and insightful comments during the preparation of this paper, and the journal reviewers for their helpful suggestions for improvement of the first draft. The research presented is partially funded by an EPSRC DTA Studentship.

REFERENCES

- [1] D. K. Roy, "Learning from sights and sounds: A computational model," Ph.D. dissertation, MIT, Media Laboratory, sep 1999.
- [2] S. Hurley and N. Chater, Eds., *Perspectives on Imitation: From Neuroscience to Social Science*. Cambridge, MA: MIT Press, 2005.
- [3] R. W. Byrne and A. E. Russon, "Learning by imitation: a hierarchical approach," *Brain and Behavioral Sciences*, vol. 21, no. 5, pp. 667–721, 1998.
- [4] J. Demiris, S. Rougeaux, G. M. Hayes, L. Berthouze, and Y. Kuniyoshi, "Deferred imitation of human head movements by an active stereo vision head," in *Proceedings of the 6th IEEE International Workshop on Robot Human Communication*. Sendai, Japan: IEEE Press, September 1997, pp. 88–93.
- [5] M. J. Matarić, "Sensory-motor primitives as a basis for imitation: Linking perception to action and biology to robotics," in *Imitation in Animals and Artifacts*, ser. Complex Adaptive Systems, K. Dautenhahn and C. L. Nehaniv, Eds. The MIT Press, 2002, ch. 15, pp. 391–422.
- [6] M. A. Wood, J. C. S. Leong, and J. J. Bryson, "ACT-R is almost a model of primate task learning: Experiments in modelling transitive inference," in *Proceedings of the Twenty-Sixth Annual Conference of the Cognitive Science Society*, K. Forbus, D. Gentner, and T. Regier, Eds., Cognitive Science Society. Chicago, Illinois, USA: Lawrence Erlbaum Associates, Inc., August 2004, pp. 1470 – 1475.
- [7] J. J. Bryson and J. C. S. Leong, "Primate errors in transitive 'inference': A two-tier learning model," *Animal Cognition*, 2006, in press.
- [8] R. A. Rensink, "The dynamic representation of scenes," *Visual Cognition*, vol. 7, pp. 17–42, 2000.
- [9] R. Dawkins, "Hierarchical organisation: A candidate principle for ethology," in *Growing Points in Ethology*, P. P. G. Bateson and R. A. Hinde, Eds. Cambridge: Cambridge University Press, 1976, pp. 7–54.
- [10] T. M. Caro and M. D. Hauser, "Is there teaching in nonhuman animals," *The Quarterly Review of Biology*, vol. 67, no. 2, pp. 151–174, June 1992.
- [11] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, ser. Adaptive Computation and Machine Learning. Cambridge, Mass.: MIT Press (Bradford Book), 1998.
- [12] D. Gordon, "APT agents: Agents that are adaptive, predictable, and timely," in *Proceedings of the First Goddard Workshop on Formal Approaches to Agent-Based Systems (FAABS'00)*, 2000.
- [13] M. Barley and H. W. Guesgen, Eds., *Spring Symposium on Safe Learning Agents*. AAAI, March 2002.
- [14] C. L. Nehaniv and K. Dautenhahn, "The correspondence problem," in *Imitation in Animals and Artifacts*, ser. Complex Adaptive Systems, K. Dautenhahn and C. L. Nehaniv, Eds. The MIT Press, 2002, ch. 2, pp. 41–61.
- [15] S. Hurley, "Active perception and perceiving action: The shared circuits model," in *Perceptual Experience*, T. Gendler and J. Hawthorne, Eds. Oxford University Press, 2005, ch. 6.
- [16] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society series B*, vol. 39, pp. 1–38, 1977.
- [17] T. Mitchell, *Machine Learning*. New York, NY: McGraw Hill, 1997.
- [18] D. K. Roy and A. P. Pentland, "Learning words from sights and sounds: a computational model," *Cognitive Science*, vol. 26, pp. 113–146, 2002.
- [19] E. D. Thiessen and J. R. Saffran, "When cues collide: Statistical and stress cues in infant word segmentation," *Developmental Psychology*, vol. 39, pp. 706–716, 2003.
- [20] R. W. Byrne, "Imitation as behaviour parsing," *Philosophical Transactions of the Royal Society B*, vol. 358, pp. 529–536, 2003.
- [21] G. L. Drescher, *Made-up minds : a constructivist approach to artificial intelligence*. MIT Press, Cambridge, MA, 1991.
- [22] A. Alissandrakis, C. L. Nehaniv, K. Dautenhahn, and J. Saunders, "Achieving corresponding effects on multiple robotic platforms: Imitating in context using different effect metrics," in *Proceedings of the Third International Symposium on Imitation in Animals and Artifacts*. University of Hertfordshire, Hatfield, UK: SSAISB, April 2005, pp. 10–19.
- [23] V. V. Hafner, "Cognitive maps in rats and robots," *Journal of Adaptive Behavior, special issue on "Towards Artificial Rodents"*, vol. 13, no. 2, pp. 87–96, 2005.
- [24] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233–242, 1999.
- [25] T. Tyrrell, "Computational mechanisms for action selection," Ph.D. dissertation, University of Edinburgh, 1993, centre for Cognitive Science.
- [26] L. S. Zetlemoyer, H. Pasula, and L. P. Kaelbling, "Learning planning rules in noisy stochastic worlds," in *AAAI*, M. M. Veloso and S. Kambhampati, Eds. Pittsburgh, PA: AAAI Press, July 2005, pp. 911–918.
- [27] J. E. Laird, "Using a computer game to develop advanced AI," *Computer*, vol. 34, no. 7, pp. 70–75, July 2001.
- [28] R. Le Hy, A. Arrigoni, P. Bessi ere, and O. Lebeltel, "Teaching bayesian behaviours to video game characters," *Robotics and Autonomous Systems*, vol. 47, pp. 177–185, 2004.
- [29] C. Thureau, C. Bauckhage, and G. Sagerer, "Imitation learning at all levels of game-AI," in *Proc. Int. Conf. on Computer Games, Artificial Intelligence, Design and Education*, 2004, pp. 402–408.
- [30] J. J. Bryson, W. Lowe, and L. A. Stein, "Hypothesis testing for complex agents," in *NIST Workshop on Performance Metrics for Intelligent Systems*, A. M. Meystel and E. R. Messina, Eds. Washington, DC: NIST Special Publication 970, August 2001, pp. 233–240.
- [31] Digital Extremes, "Unreal Tournament," 1999, epic Games, Inc.
- [32] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Philosophical Transactions of the Royal Society of London, B*, vol. 358, no. 1431, pp. 537–547, 2003.

- [33] E. Bizzi, S. Giszter, E. Loeb, F. A. Mussa-Ivaldi, and P. Saltiel, "Modular organization of motor behavior in the frog's spinal cord," *Trends in Neuroscience*, vol. 18, pp. 442–446, 1995.
- [34] M. S. A. Graziano, C. S. R. Taylor, T. Moore, and D. F. Cooke, "The cortical control of movement revisited," *Neuron*, vol. 36, pp. 349–362, October 2002.
- [35] D. Cusance, A. Whiten, and T. Fredman, "Social learning of an artificial fruit task in capuchin monkeys (*cebus apella*)," *Journal of Comparative Psychology*, vol. 113, no. 1, pp. 13–23, March 1999.
- [36] Digital Extremes, "UnrealEd 2.0," 2000, epic Games, Inc.
- [37] A. von Stein and J. Sarnthein, "Different frequencies for different scales of cortical integration: From local gamma to long range alpha-theta synchronization," *International Journal of Psychophysiology*, vol. 38, no. 301–313, 2000.
- [38] S. Schaal and C. G. Atkeson, "Robot juggling: An implementation of memory-based learning," *Control Systems Magazine*, vol. 14, no. 1, pp. 57–71, 1994.
- [39] J. J. Bryson and B. McGonigle, "Agent architecture as object oriented design," in *The Fourth International Workshop on Agent Theories, Architectures, and Languages (ATAL97)*, M. P. Singh, A. S. Rao, and M. J. Wooldridge, Eds. Providence, RI: Springer-Verlag, 1998, pp. 15–30.
- [40] K. R. Thórisson, "A mind model for multimodal communicative creatures & humanoids," *International Journal of Applied Artificial Intelligence*, vol. 13, no. 4/5, pp. 519–538, 1999.
- [41] J. J. Bryson, "Modular representations of cognitive phenomena in AI, psychology and neuroscience," in *Visions of Mind: Architectures for Cognition and Affect*, D. N. Davis, Ed. Idea Group, 2005, pp. 66–89.
- [42] J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly, "Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory," *Psychological Review*, vol. 102, no. 3, pp. 419–457, 1995.
- [43] N. R. Carlson, *Physiology of Behavior*. Boston: Allyn and Bacon, 2000.
- [44] K. Louie and M. A. Wilson, "Temporally structured replay of awake hippocampal ensemble activity during rapid eye movement sleep," *Neuron*, vol. 29, no. 1, pp. 145–156, Jan 2001.
- [45] T. J. Teyler and P. Discenna, "The hippocampal memory indexing theory," *Behavioral Neuroscience*, vol. 100, pp. 147–154, 1986.
- [46] W. Lowe, "Semantic representation and priming in a self-organizing lexicon," in *Proceedings of the Fourth Neural Computation and Psychology Workshop: Connectionist Representations (NCPW4)*, J. A. Bullinaria, D. W. Glasspool, and G. Houghton, Eds. London: Springer-Verlag, 1997, pp. 227–239.
- [47] T. Kobayashi, H. Nishijo, M. Fukuda, J. Bures, and T. Ono, "Task-dependent representations in rat hippocampal place neurons," *Journal of Neurophysiology*, vol. 78, no. 2, pp. 597–613, 1997.
- [48] J. E. Laird and P. S. Rosenbloom, "The evolution of the Soar cognitive architecture," in *Mind Matters*, D. M. Steier and T. M. Mitchell, Eds. Erlbaum, 1996.
- [49] J. R. Anderson and M. Matessa, "The rational analysis of categorization and the ACT-R architecture," in *Rational Models of Cognition*, M. Oaksford and N. Chater, Eds. Oxford University Press, 1998.
- [50] G. Rizzolatti, L. Fogassi, and V. Gallese, "Cortical mechanisms subserving object grasping and action recognition: A new view on the cortical motor functions," in *The New Cognitive Neurosciences*, 2nd ed., M. S. Gazzaniga, Ed. Cambridge, MA: MIT Press, 2000, ch. 38, pp. 538–552.
- [51] J. J. Bryson and M. A. Wood, "Learning discretely: Behaviour and organisation in social learning," in *Third International Symposium on Imitation in Animals and Artifacts*, Y. Demiris, Ed. Hatfield, UK: The Society for the Study of Artificial Intelligence and the Simulation of Behaviour, April 2005, pp. 30–37.
- [52] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1–2, pp. 99–134, 1998.
- [53] S. Benson, "Learning action models for reactive autonomous agents," Ph.D. dissertation, Stanford University, December 1996, department of Computer Science.
- [54] A. Whiten, "Imitation of the sequential structure of actions by chimpanzees (*pan troglodytes*)," *Journal of Comparative Psychology*, vol. 112, pp. 270–281, 1998.
- [55] W. T. Fitch, "The evolution of speech: A comparative review," *Trends in Cognitive Sciences*, vol. 4, no. 7, pp. 258–267, 2000.



Mark Wood Biography text here.



Joanna Bryson Biography text here.