2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

A Game-Theoretic Procedure for Learning Hierarchically Structured Strategies

Benjamin Rosman and Subramanian Ramamoorthy

Abstract— **This paper addresses the problem of acquiring a hierarchically structured robotic skill in a nonstationary environment. This is achieved through a combination of learning primitive strategies from observation of an expert, and autonomously synthesising composite strategies from that basis. Both aspects of this problem are approached from a game theoretic viewpoint, building on prior work in the area of multiplicative weights learning algorithms. The utility of this procedure is demonstrated through simulation experiments motivated by the problem of autonomous driving. We show that this procedure allows the agent to come to terms with two forms of uncertainty in the world – continually varying goals (due to oncoming traffic) and nonstationarity of optimisation criteria (e.g., driven by changing navigability of the road). We argue that this type of factored task specification and learning is a necessary ingredient for robust autonomous behaviour in a "large-world" setting.**

I. INTRODUCTION

Autonomous robots must operate in a "large-world" setting where information is necessarily incomplete and it is infeasible for the designers to *ex ante* anticipate all contingencies that the agent might face. It is generally believed that many such behaviours admit a hierarchical or layered structure [1], [2] and many successful examples of complex engineered systems have leveraged this feature [3]. To the extent that a truly autonomous system must learn to acquire and adapt its behaviours online, there is a need for algorithmic techniques that enable efficient learning of such hierarchical strategies – from a combination of expert advice and direct experience. It is generally acknowledged that humans learn better by means of a curriculum [4], as a fast and safe method of skill acquisition [5]. The question for us is that of enabling machines to accomplish the same, to provide a measure of robustness to noise and an ability to handle larger scale changes due to nonstationarities.

The former problem, that of learning from an expert, has been studied by a number of research groups in recent years. An excellent state-of-the-art example is the apprenticeship learning algorithm of Abbeel and Ng [6]. In this setting, one utilizes an expert, with access to a sophisticated control policy for the desired task, to provide the target for a learning agent. The apprentice agent may be equipped with a set of elementary actions and is required to approximate the expert's policy using that basis.

This work was supported by the Skye Foundation.

B. Rosman is with the School of Informatics, University of Edinburgh, Edinburgh, UK B.S.Rosman@sms.ed.ac.uk

S. Ramamoorthy is with the School of Informatics, University of Edinburgh, Edinburgh, UK s.ramamoorthy@ed.ac.uk

This notion of imitation learning actually has a long and rich history. For instance, the problem has been looked at from a social and psychologically inspired angle in [7], [8] and [9]. Schaal et al. [10] discuss several fundamental issues in imitation learning, such as the use of action primitives. Billard et al. [11] use Hidden Markov Models and focus on the determination of what to imitate in order to learn to trace the shapes of letters. Inamura et al. [12] also use Hidden Markov Models to learn various motions. A majority of these approaches pose the problem of imitation learning as the statistical learning problem of approximating the behaviour of the expert, or perhaps by posing the problem as one of learning the reward function in a reinforcement learning setting [13]. However, in realistic scenarios, expert demonstrations must necessarily be incomplete. Moreover, one is interested in acquiring skills in a large-world sense that includes, in addition to the basic policy for task execution with robustness to noise, variations for handling unanticipated contingencies in a nonstationary world. This is the focus of the work presented in this paper.

The problem of operating in a stochastic environment has been previously considered in the form of robust control of Markov Decision Processes [14] which considers the problem in a game theoretic sense, as we also do. However, the complexity of such solution methods can easily render them intractable in real-world problems. Instead, we adopt a factored, hierarchical approach to defining skills. Accordingly, we also take a somewhat different approach to learning the solutions to the games. This problem has also been considered by detecting a changing context in which an agent is operating, and changing policies appropriately [15]. Again this approach develops flat, non-hierarchical strategies, and does not address the reuse of primitive policies as we would like to be able to do.

The apprenticeship learning procedure in [6] begins to address the issue of task variations by learning the reward function (i.e., *inverse* reinforcement learning) so that subsequent variations may be handled separately by the agent. This work was extended in [16] to address the issue of disturbances more directly, by adopting a game theoretic approach. They propose the Multiplicative Weights Algorithm for Apprenticeship Learning (MWAL), which poses the policy approximation problem in a minimax setting – allowing the agent to find the best approximation (within a class of primitives that may well differ from that of the expert) to the demonstrated behaviour that is the target.

We take a layered approach to skill acquisition. Firstly, we utilise expert demonstration to learn a set of primitive

skills that will form the basis for subsequent autonomous learning. In this paper, we use the MWAL algorithm to solve this aspect of the problem, although we note that other choices would not impact the subsequent steps. With this basis, we pose the problem of action synthesis in a nonstationary world as one of playing a game against nature, where nature picks variations of the problem (e.g., different optimality criteria – such as changes in navigability of the road) and the agent must learn to synthesize the optimal response in terms of a mixed strategy over previously learned primitives. We again solve this second stage problem using a multiplicative weights algorithm. In this setting, the role of the expert is to bootstrap the learning agent from a state of complete ignorance of the skill to be learned, to one of partial knowledge from which complex skills are learnable.

The structure of this paper is as follows. Section II describes the basic MWAL procedure and our modified algorithm that addresses the hierarchical skills setting. The performance of this algorithm is demonstrated through empirical experiments described in section III. This experiment is a modified version of the driving task from Abbeel and Ng [6] – elaborated in order to incorporate two different sources of task variations (changing terrain variability, impacting the cost structure of the optimisation, and changes within the oncoming traffic which forms the basic task). The main result is the demonstration that the learning agent has efficiently acquired a composite strategy for operation in a nonstationary scenario consisting of nontrivial variations of the task that the expert *has not* demonstrated before.

Although most of the discussion in this paper is focussed on this specific application scenario, of vehicle navigation, the underlying algorithmic ideas are more generally applicable to problems of robot motion [17]. A general description of robot motion involves the generation of trajectories over an abstract surface, i.e., the configuration space. Motion in this space is dictated by laws of motion that impact the structure of the metric on this space, such that different local regions correspond to different levels of 'navigability'. Many realistic robotics problems, e.g., full-body humanoid behaviours, are such that relatively small changes in the external environment could result in much larger changes in this underlying cost structure – this is what makes generalisation of acquired skills hard. Moreover, the goal of robotics is for robots to perform interesting manoeuvres, under a variety of different circumstances and in response to changing goals. So, this is a necessary and intrinsic feature of all problems in robotics. A higher-dimensional extension of our driving scenario to this abstract space would still result in similar algorithmic questions and, we believe, could be solved as natural extensions to the procedure outlined in this paper.

II. ARCHITECTURE OF THE SKILL ACQUISITION FRAMEWORK

We view the skill acquisition process as a game between the learning agent and an adversarial environment. In a game theoretic sense, this corresponds to an optimisation of the expected rewards received by two interacting agents. This is done by changing weighted distributions over the different strategies which can be used by the players. The optimal mixing of strategies for both players in this way is known as a Nash equilibrium. We assume the environment is controlled by an unknown process – nature. The learning agent is required to synthesise complex manoeuvres, from a basis of simple ones, which are learned from an expert, as it is unrealistic to assume that the expert could demonstrate behaviour for every conceivable scenario, particular if the environment is dynamic, contains other agents, or is prone to noise.

Our proposed skill acquisition framework of primitive strategy learning, followed by composite strategy synthesis is shown in Figure 1. The synthesis is by regret minimisation [18] to find a Nash equilibrium between the agent's policies, and adversarial changes to the cost structure presented by the environment.

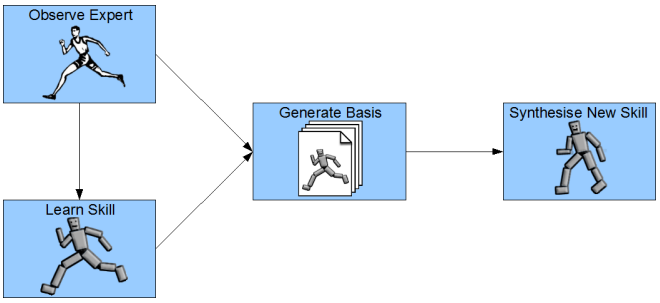

Fig. 1. The proposed skill acquisition framework. Observing and learning from an expert allows the agent to acquire primitive skills. Once a library of these primitive skills has been assembled, they can be synthesised to generate novel variants of the skill which are suitable for conditions that may differ from those initially demonstrated by the expert.

The composite strategy is the equilibrium in the game between the player, who is looking for a mixed strategy defined over primitives, and nature which is altering the cost/reward structure of the optimal control/dynamic game problem – while, the entire time, there is also the variation in the goal or target with which the agent is working. In general, the game theoretic formulation allows one to be more sophisticated about modelling the structure of the adversarial changes than alternative approaches that may, e.g., collapse the unknowns into a (stationary) latent variable or partially observable state. Such formulations also allow us to draw on a rich literature on learning in the presence of adversaries that need not be fully modelled, e.g., see [19].

While we use regret minimisation for solving these games, we note that this is merely a tool which could be swapped out for others, such as a dynamic programming [14], based on computational or other considerations. However, the notion of mixture distributions over strategies allows us more flexibility in expressing composite strategies.

A. Learning Primitive Strategies

The objective of the agent is to learn a policy $\pi(s, a)$ which maps states to a probability distribution over the actions in each state. The problem of learning such a policy

as it is posed here is one of reinforcement learning. However, in this case the true reward function $R^*(s)$ is unknown, and instead the learning agent has a vector of features $\phi : \mathcal{S} \mapsto [-1, 1]^k$ which describes the desirability of particular aspects of a state. The true reward function is assumed to be a linear combination of these features $R^*(s) = w^* \cdot \phi(s)$ where $w^* \in \Re^k$ is an unknown weight vector.

This weight vector can be learned in a number of ways. Here, we use the MWAL algorithm [16]. This approach defines a feature expectations vector as the expected, cumulative, discounted feature values of a policy π as $\mu(\pi) = E[\sum_{t=0}^{\infty} \gamma^t \phi(s_t)|\pi, \Theta, D]$. Given an expert's policy π_E, the goal is to find a policy π for that agent, such that $V(\pi) \geq V(\pi_E) - \epsilon$, where $V(\pi) = E[\sum_{t=0}^{\infty} \gamma^t R^*(s_t)|\pi, \Theta, D]$ is the value of a policy π on the MDP M, and thus $V(\pi) = w^* \cdot \mu(\pi)$.

The MWAL algorithm views the decision making process as a two player zero-sum game, where the maximising player is the learning agent selecting a distribution over the policies, and the minimising player is the environment which selects the reward function through the values of w. The agent wishes to find a mixed policy ψ to maximise $V(\psi) - V(\pi_E)$ with unknown and potentially worst-case choice of w. This optimisation against a worst case choice of w represents the fact that the true underlying behaviour of the environment is unknown, and the results of the agent's interactions with the environment are approximated by the feature vectors. As the apprentice learns exclusively from the feature expectations of the expert μ_E, the skill acquisition of the apprentice is independent of not only the control policy used by the expert, but also its state and action space. The apprentice may thus only have an approximation of the actions available to the expert, yet still be able to learn the demonstrated skill. This problem is posed as a game, represented as

$$v^* = \max_{\psi \in \Psi} \min_{w} w^T \mathbf{G} \psi \tag{1}$$

where $\mathbf{G}(i, j) = \mu^j(i) - \mu_E(i)$, $\mu_E = \mu(\pi_E)$, $\mu^j = \mu(\pi^j)$ for π^j the j^{th} stationary policy, and $\mu(i)$ is the i^{th} component of μ.

The MWAL Algorithm (Alg. 1) solves this game using the regret minimisation procedure of iteratively adjusting the weights w, using a multiplicative weights algorithm for solving large games [20] and then finding an optimal policy ψ for the current weights using an on-policy Monte Carlo control algorithm [21].

The output of this algorithm is a mixed policy, using which the agent can perform almost as well as, if not better than, the expert on the given task and domain, in terms of the reward function composed of the feature vectors.

B. Learning Composite Strategies

The composite strategy synthesis problem involves a game, where the agent is to maximise the reward gained from the MDP in novel task instances given by nature. As the behaviour of the environment is unknown, optimisation is against possible worst-case instances of the game, as specified by the combination of goals and reward structure.

Algorithm 1: The MWAL Algorithm (from [16])

Input: An MDP M, and an estimate of the feature expectations of the expert, $\hat{\mu}_E$
Output: A mixed policy $\bar{\psi} = \{\hat{\pi}^{(t)}\}$ for all t
begin
  $\beta \longleftarrow \left(1 + \sqrt{\frac{2 \ln k}{T}}\right)^{-1}$
  $\mathbf{W}^{(1)}(i) \longleftarrow 1$ for all $i = 1, \ldots, k$
  $\mathbf{G}(i, \mu) \longleftarrow ((1 - \gamma)(\mu(i) - \hat{\mu}_E(i)) + 2)/4$
  for $t = 1, \ldots, T$ **do**
    $\mathbf{w}^{(t)}(i) \longleftarrow$ normalised $\mathbf{W}^{(t)}(i)$ for all i
    Compute $\hat{\pi}^{(t)}$ w.r.t. $R(s) = \mathbf{w}^{(t)} \cdot \phi(s)$
    Estimate $\hat{\mu}^{(t)} \approx \mu(\hat{\pi}^{(t)})$
    $\mathbf{W}^{(t+1)}(i) \longleftarrow \mathbf{W}^{(t)}(i) \beta^{\mathbf{G}(i, \hat{\mu}^{(t)})}$ for all i
  end
  $\bar{\psi} \longleftarrow \{\hat{\pi}^{(t)}\}$ for all t with probability $\frac{1}{T}$
  return *mixed policy* $\bar{\psi}$
end

Consider a two-player repeated matrix game $\mathbf{G}$ where the agent is the row player aiming to maximise its reward in using its skill in the presence of the column player, nature. The actions of the agent are the primitive policies learned in Algorithm 1, $\Psi = \{\psi_1, \psi_2, \ldots, \psi_N\}$, and those of nature are $\mathbf{Q} = \{Q_1, Q_2, \ldots, Q_M\}$. The set $\mathbf{Q}$ consists of behaviours which could be executed by the environment, such as structural changes to the world or movements made by strategic adversaries. The value of the game matrix for using policies (ψ_i, Q_j) is $\mathbf{G}(i, j)$ and is dependent on the parameters of the new scenario. The required solution is a Nash equilibrium (ψ^*, Q^*) [22].

This mixed strategy equilibrium is found using regret minimisation by means of a multiplicative weights algorithm [16], [20]. The Strategy Composition Algorithm is shown in Algorithm 2. This differs from Algorithm 1 in the way we compute $\mathbf{G}$. $E[R(P, Q)]$ is the expected reward of policy P in the presence of nature's policy Q. So the value of the game is computed as a normalised difference between the game values obtained by the current mixed policy, and any of the primitive policies. This allows the weighting to favour the more successful of these policies.

The algorithm starts with a mixed policy consisting of a uniform distribution over the primitive policies, and uses an iterative procedure to update this distribution by multiplicatively allocating more weight to the policies which perform better, as determined by the game value. The output is a mixed policy consisting of a distribution over the primitive policy set Ψ giving an equilibrium strategy.

This composed strategy $\tilde{\psi}$ is a linear combination of the policies in Ψ. Assume strategy ψ_i selects an action a_j in state s_k with probability p^i_{kj}, and the probability of being in state s_k under policy ψ_i is p^i_k. Then, given a final mixed strategy weighting $\mathbf{w}^{(T+1)} = \{w_1, w_2, \ldots, w_N\}$ with $\sum_i w_i = 1$, the policy $\tilde{\psi}$ would choose action a_j in state s_k with probability $\tilde{p}_{kj} = \sum_i w_i p^i_{kj} p^i_k$. The weights are biased towards the policies that achieve a higher expected reward. Using regret

Algorithm 2: The Strategy Composition Algorithm

Input: An MDP $\tilde{M}$, and a set of primitive strategies $\Psi = \{\psi_1, \psi_2, \ldots, \psi_N\}$
Output: A mixed policy $\tilde{\psi} = \mathbf{w} \cdot \Psi$
begin
$\beta \longleftarrow \left(1 + \sqrt{\frac{2 \ln N}{T}}\right)^{-1}$
$\mathbf{w}^{(1)}(i) \longleftarrow \frac{1}{N}$ for all $i = 1, \ldots, N$
for $t = 1, \ldots, T$ **do**
$\tilde{\psi} = \mathbf{w}^{(t)} \cdot \Psi$
$\mathbf{G}(\Psi, \tilde{\psi}, \mathbf{Q}) \longleftarrow$
$((E[R(\tilde{\psi}, \mathbf{Q})] - E[R(\Psi, \mathbf{Q})]) + 1)/2$
$\mathbf{W}(i) \longleftarrow \mathbf{w}^{(t)}(i)\beta^{\mathbf{G}(\psi_i, \tilde{\psi}, \mathbf{Q})}$ for all i
$\mathbf{w}^{(t+1)}(i) \longleftarrow \frac{\mathbf{W}(i)}{\sum_j \mathbf{W}(j)}$ for all $i = 1, \ldots, N$
end
return *mixed policy* $\tilde{\psi} = \boldsymbol{w}^{(T+1)} \cdot \Psi$
end

minimisation, Algorithm 2 thus learns to rely more heavily on the strategies which generate higher rewards on this unseen MDP. We note that this probability is difficult to compute analytically, but is approximated by convergence of the algorithm, which ultimately results in a probability distribution over the actions in every state.

An alternative to this approach would have been to create a distribution over possible world states, estimate the current state given sensory evidence and apply suitable policies. However, we assume much less *a priori* knowledge of contingencies and a more severe level of on-line variations where direct state estimation from limited data would be problematic.

III. EXPERIMENTAL RESULTS

A. Experimental Setup

Experiments were run on a vehicle navigation domain, of a car learning to navigate through traffic on a multi-lane one-way road. All other traffic moves at a fixed speed, slower than the agent. The other cars do not exhibit any complicated behaviour such as changing lanes. The road is discretised by car length, so the road is a two-dimensional grid, where the length is the number of car lengths, and the width is the number of lanes. The road is either paved or not at each point, with a high and low reward for using such a portion of the road respectively. This could be generalised for intermediate levels of navigability.

There are two different sources of randomness encountered by the agent. The first is the layout of the terrain over which the agent drives, which is a parameter corresponding to different road conditions (equivalent to the cost function in an optimal control problem). The second is other cars on the road, acting as variations/adversarial disturbances to the basic driving forward task, generated by nature by means of some unknown process. The agent must learn to drive optimally on a given terrain in these adversarial traffic conditions, where optimality is defined by its attempts to maximise time spent on the paved sections of the road whilst avoiding collisions.

The avoidance actions, of moving to the left or the right, taken by the agent are dependent only on the cars in some finite horizon. If a collision with a car is imminent, it is irrelevant to the agent whether or not there are other cars behind that car. Thus only the distance to the closest car in each lane is recorded.

The expert navigates the road using a set of *if-then* rules with some additional random effects. This could just as easily have been an optimal controller, or a human controller.

The learning agent is presented with a basis of scenarios on which to learn behaviours from the expert, to then synthesise them to form strategies for unseen terrains. The basis was selected as a set of intuitive road shapes, but is by no means an 'optimal' basis for all eventualities. In general, we claim that it is unrealistic to expect that a complete and comprehensive basis for all complex scenarios that could be encountered in a skill could be enumerated. The set of basis roads is shown in Figure 2. Let these basis roads be denoted $R1$ through $R8$.

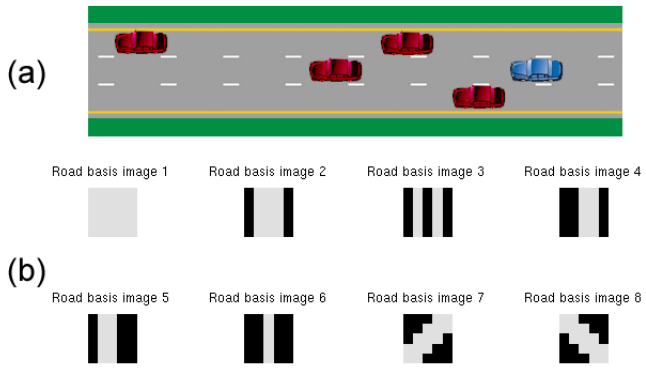

Fig. 2. (a) An example set up of the road navigation problem, with three of five lanes paved [6]. Note, the grass areas are unpaved lanes. (b) The set of basis roads used in the experiments, all sized 5×5: each consists of five lanes, of five car lengths each. The dark areas depict unfavourable 'unpaved' regions, and the light areas show the 'paved' regions of higher reward.

B. Results

The primitive strategy learning stage involves learning optimal strategies for performing a skill in a known environment from the observation of the performance of an expert. In each iteration, the starting location for the agent was drawn uniformly from the five lanes.

An example of the first experiment is depicted in Figure 3. In this case it was obviously not possible for even the expert to avoid both collisions and leaving the paved lanes as there were two separated single lanes. Thus, although the vast majority of time is spent in lanes 2 and 4, all of the other three lanes were marginally used, in order to avoid collisions. The agent learned a very similar distribution to the expert, although the number of collisions was slightly higher, as the Monte Carlo policy iteration uses soft policies, meaning

that a small chance of choosing an alternate policy over the currently determined best one remains. This is because a collision and deviating from the road are considered equally bad.

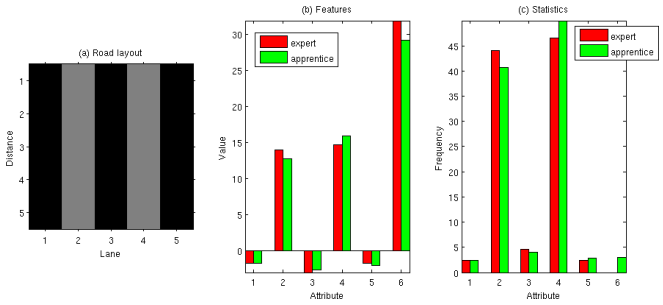

Fig. 3. Primitive Policy for $R3$. (a) primitive road on which the apprentice is being trained: a split road. (b) the six features for that policy ($\mu(\pi)$) where the first five attributes correspond to the usage of each lane, and the sixth is a lack of collisions. (c) the lane use statistics and collisions of the final trained policy when being executed for 100 iterations on that road.

Figure 4 demonstrates results of synthesising primitive strategies into composite policies, as a game between the learning agent and nature. The actions available to the learning agent are the policies from the basis terrains. The car generation policies of nature, Q, are randomly selected at each time step from a set of predefined policies, such as "generate a car on a random lane", "generate a car on a random paved lane" or "generate two cars". The value of a game between these two players is the difference of the sums of their feature expectations on that terrain. This provides a scalar value for the game as $\mathbf{G}(\psi, Q)$ for the agent using a strategy ψ and the environment using strategy Q.

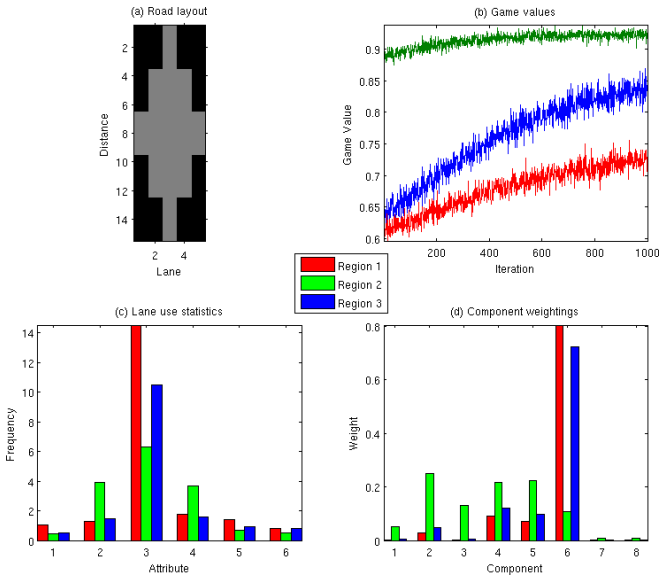

Fig. 4. New Scenario Experiment. (a) the layout of the road. (b) change in game values over the course of 1,000 iterations of the experiment. (c) lane usage statistics at the final iteration: the five lanes are indicated by attributes 1 through 5, and attribute 6 shows the number of collisions. (d) the weights assigned to each element of the basis for constructing the final policy.

The roads are divided into regions, which are 5×5 patches, i.e. sections of road consisting of five lanes, each of five car lengths. This provides the algorithm with a predefined finite horizon for implementing a particular mixture of strategies.

Regions 1 and 3 are similar in shape, and this is recognised by the algorithm which devises a similar mixture of policies for these two regions. The fact that region 2 has unpaved areas only at the four corners means there is only limited scope for learning, reflected in the fact that there is only a very gradual increase in the game value. The improvement is more marked in the blue and red graphs for regions 1 and 3 – the areas with considerable narrowing.

Another example of the composite strategy synthesis is shown in Figure 5, as a long meandering road. Again, performance in every region improves with more iterations. Many interesting features can be observed in the composite strategy, such as the fact that regions 1 and 3 rely heavily on the policy from $R4$, while regions 2 and 4 use the policy from $R5$. This corresponds to the road repeatedly alternating between swinging to the left and to the right.

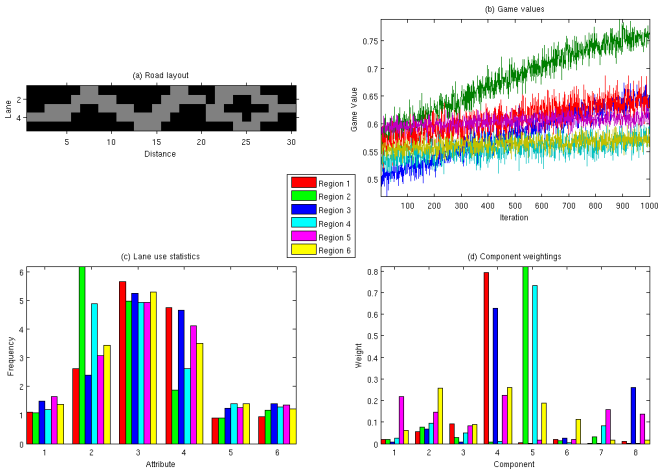

Fig. 5. New Scenario Extended Experiment. (a) the layout of the road. (b) change in game values over the course of 1,000 iterations of the experiment. (c) lane usage statistics at the final iteration: the five lanes are indicated by attributes 1 through 5, and attribute 6 shows the number of collisions. (d) the weights assigned to each element of the basis for constructing the final policy.

C. Assessment

The performance of the synthesised strategies was assessed as follows: when running a policy on a road, two averages were generated – the average number of times the policy deviated from paved road per time step (the accuracy), and the average number of collisions per time step. Both the full contingent of eight primitive basis policies, and the composite policy trained specifically for that road were run on each of the eleven roads in Figure 6. The average accuracy and collisions per time step were calculated for each policy.

Figure 7 plots the accuracy of the nine policies, as well as the average of the eight primitive policies. As can be seen, the mixed composite policy acts as an approximate bound on the performance of the primitive policies, and is considerably better in accuracy than the average of the primitives. This

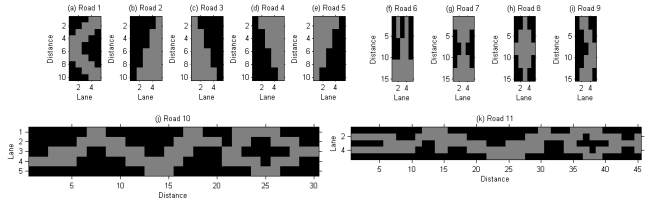

Fig. 6. Experiment Road Set. The first five are short templates, sized 10×5. Roads (f) to (i) are medium sized at 15×5, and roads (j) and (k) are long roads, of length 30 and 45 respectively.

shows that the composite strategy is consistently among the candidates for most accurate policy.

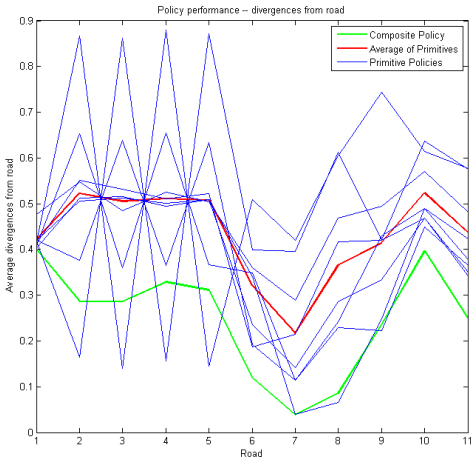

Fig. 7. Policy accuracy over 4,500 time steps. The x-axis represents the different roads sorted by increasing length. As these are deviations from the paved section, lower values are better.

The shorter roads show the composite policy to perform better than the majority of the primitive policies, the medium roads show a similar accuracy between the composite and primitive policies, and as the roads lengthen, the composite policies improve compared to the primitives and ultimately outperform them. The reason is that shorter road templates are more likely to closely match one of the primitive roads. If one of these primitive policies outperforms the other primitives on both regions of a two-region road, the composite would learn to be based predominantly on that policy, and may still be outperformed by that policy. A longer road decreases the chance of one of the basis policies performing well continuously, providing the composite policy with an opportunity to draw on elements of different components of the basis at various times for improved results.

The second performance metric is the average number of collisions per time step, displayed in Figure 8. This figure shows three 'bands' of results.

The primitives in the 'lower' band are policies corresponding to $R1$, $R2$, $R4$ and $R5$. These are the 'broad' policies, having two or more adjacent paved lanes, meaning the policy is less likely to have to leave the road to avoid a collision. As collisions and deviating from the paved road are equally

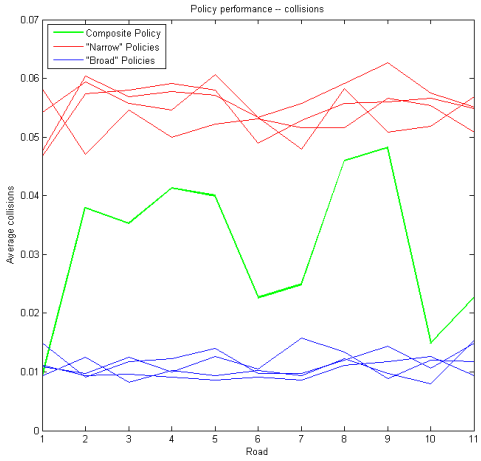

Fig. 8. Policy collisions over 4,500 time steps. The x-axis represents the different roads sorted by increasing length. Lower values are better.

undesirable, these policies will tend to have fewer collisions as it is easier for them to avoid collisions without incurring terrain-based costs. Conversely, the policies contributing to the 'higher' band are those corresponding to $R3$, $R6$, $R7$ and $R8$. These are the 'narrow' policies, not consisting of segments of several adjacent lanes. Avoiding collisions is likely to involve leaving the paved areas of road and incurring penalties. Consequently, the number of collisions is higher for these policies.

The composite policy lies between these bands. The margin between the 'higher' and 'lower' band is small and the number of collisions of the composite policy is bounded by the primitive policies. Clearly there are trade-offs between accuracy and collisions in the creation of the policies, and is a consequence of the fact that neither property is indicated to the agent to be better or worse.

The task to be completed is arbitrary, as no assumptions are made other than that the expert has some (not necessarily optimal) strategy for performing that task. Repeated learning of a skill in different scenarios with the Multiplicative Weights for Apprenticeship Learning (MWAL) Algorithm establishes a basis of strategies. A combination of these is learned, to give a control policy for performing that skill in a new setting with arbitrary disturbance processes. The two phases of this algorithm provide a mechanism whereby the learning agent is guided through the acquisition of a basic set of skills by a teaching agent, and then extrapolates these skills to more sophisticated settings with an unknown adversarial environment.

IV. DISCUSSION AND CONCLUSIONS

This work is motivated by the desire to create autonomous agents that can synthesise near-optimal actions in a non-stationary world – involving variations in both the goals and the cost structure of the optimal control/dynamic game problem – seeded by a certain level of expert advice in the form of demonstrations in the same domain. With this

in mind, we have presented a procedure that combines the notion of apprenticeship learning (or inverse reinforcement learning) with the notion of action synthesis by solving a game against nature (using a multiplicative weights procedure). Our experimental results demonstrate our claims in the domain of vehicle navigation, but this concept of skill synthesis under adversarial conditions could be easily lifted to higher dimensions.

This is a first step in a program of research aimed at understanding the process of continual skill acquisition in an open-ended world. The need for this sort of flexibility and resilience has been noted by a number of researchers, e.g., [23]. However, oftentimes, discussions of robustness are focussed on variations within the robot system. Our focus, in current and future work in this direction, is on coming to terms with variations in the external world (in terms of robustness to noise, structural changes in the world, and strategic adversaries) which is sometimes a much larger class, requiring different notions and models of interaction and strategy.

REFERENCES

[1] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G. Pappas, "Symbolic planning and control of robot motion," *Robotics & Automation Magazine, IEEE*, vol. 14, no. 1, pp. 61–70, 2007.

[2] H. A. Simon, *The Sciences of the Artificial*, 3rd ed. MIT Press, 1996.

[3] P. Varaiya, "A question about hierarchical systems," in *System Theory: Modelling, Analysis and Control*, T. Djaferis and I. Schick, Eds., 2000.

[4] J. Elman, "Learning and development in neural networks: The importance of starting small," *Cognition*, vol. 48, no. 1, pp. 71–99, 1993.

[5] A. Meltzoff, P. Kuhl, J. Movellan, and T. Sejnowski, "Foundations for a new science of learning," *Science*, vol. 325, pp. 284–288, 2009.

[6] P. Abbeel and A. Ng, "Apprenticeship learning via inverse reinforcement learning," *Proceedings of the International Conference on Machine Learning*, 2004.

[7] C. Breazeal and B. Scassellati, "Robots that imitate humans," *Trends in Cognitive Sciences*, vol. 6, no. 11, pp. 481–487, Nov. 2002.

[8] T. Kuriyama and Y. Kuniyoshi, "Acquisition of human-robot interaction rules via imitation and response observation," *Proceedings of the 10th international conference on Simulation of Adaptive Behavior*, pp. 467–476, 2008.

[9] Y. Demiris and M. Johnson, "Distributed, predictive perception of actions: a biologically inspired robots architecture for imitation and learning," *Connection Science*, vol. 15, no. 4, pp. 231–243, December 2003.

[10] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Philosophical Transactions of the Royal Society B*, vol. 358, pp. 537–547, 2003.

[11] A. Billard, Y. Epars, S. Calinon, S. Schaal, and G. Cheng, "Discovering optimal imitation strategies," *Robotics and Autonomous Systems*, vol. 47, pp. 69–77, 2004.

[12] T. Inamura, Y. Nakamura, I. Toshima, and H. Tanie, "Embodied symbol emergence based on mimesis theory," *The International Journal of Robotics Research*, vol. 23, no. 4-5, pp. 363–377, 2004.

[13] B. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, pp. 469–483, 2009.

[14] A. Nilim and L. El Ghaoui, "Robust control of markov decision processes with uncertain transition matrices," *Operations Research*, vol. 53, no. 5, pp. 780–798, September-October 2005.

[15] B. C. da Silva, E. W. Basso, A. L. C. Bazzan, and P. M. Engel, "Dealing with non-stationary environments using context detection," *Proceedings of the 23rd international conference on Machine learning*, pp. 217–224, 2006.

[16] U. Syed and R. Schapire, "A game-theoretic approach to apprenticeship learning," *Advances in Neural Information Processing Systems*, 2008.

[17] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms and Implementations*, 1st ed. MIT Press, 2005.

[18] N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, *Algorithmic Game Theory*, 1st ed. Cambridge University Press, 2007.

[19] D. Foster and H. Young, "Regret testing: learning to play Nash equilibrium without knowing you have an opponent," *Theoretical Economics*, vol. 1, pp. 341–367, 2006.

[20] Y. Freund and R. Schapire, "Adaptive game playing using multiplicative weights," *Games and Economic Behavior*, vol. 29, pp. 79–103, 1999.

[21] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, 1st ed. MIT Press, 1998.

[22] T. Basar and G. Olsder, *Dynamic Noncooperative Game Theory*, 2nd ed. Society for Industrial and Applied Mathematics, 1999.

[23] J. Bongard, V. Zykov, and H. Lipson, "Resilient machines through continuous self-modeling," *Science*, vol. 314, pp. 1118–1121, 2006.