Simulating the Navigation and Control of Autonomous Agents *

Erol Gelenbe and Varol Kaptan
Department of Electrical and Electronic Engineering
Imperial College
London SW7 2BT, UK
{e.gelenbe,v.kaptan}@imperial.ac.uk

Khaled Hussain
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816 USA
khaled@cs.ucf.edu

Abstract – *While traditional data fusion started with systems which exploit the output of multiple sensors so as to optimise the characterisation or recognition of objects of interest, modern information fusion systems will increasingly integrate all types of information, including behavioural information and information resulting from modelling, analysis and computation. In many critical applications, modelling the behaviour of groups of coordinated autonomous entities must be carried out within physically accurate settings in order to provide realistic information about their likely behaviour. The simulated entities must conduct autonomous actions which are realistic, which follow plans of action, but which also exhibit intelligent reactive behaviour in response to unforeseen conditions. In this paper we describe how a complex and simulation environment can be used to fuse information about the behaviour of groups of objects of interest. The fused information includes the objects' individual pursuits and aims, the physical and geographic setting within which they act, and their collective social behaviour. The group control algorithms combine reinforcement learning, social potential fields and imitation. We summarise the design of a simulation system that we have designed based on these principles.*

Keywords: Simulation, Navigation, Group Behaviour and Goals, Intelligent Agents, Robotics, Machine Learning

1 Introduction

Discrete event simulation is widely used to model, evaluate and explore operational contexts of real systems under varying synthetic conditions. Simulation runs can predict the capabilities and limitations of different operational rules or of different combinations of tactical assets. Traditionally, discrete event simulation has concentrated on the algorithmic description and control of synthetic entities which are being modelled as they accomplish some meaningful function, and simulation research has devoted much attention to appropriate workload representation and output data analysis. Less attention has been paid to the design of simulation systems in which individual animated objects (such as manned or robotic vehicles, or human individuals) are provided with broad goals (such as "go quickly to that hill, and do not get killed") and are then allowed to dynamically attain the objective based on individual adaptation and learning [3, 4, 5].

However, simulation is also a sophisticated manner of fusing information based on multiple and diverse sources, such as the physical characteristics of the objects of interest, the physical or geographic environment in which they exist and act, the goals and intentions of the different objects being considered, and their social or physical interactions.

The purpose of this paper is to report some recent results on this line of research. We consider how a variety of adaptive paradigms, including reinforcement learning, social potential fields and imitation, can be used in a simulation to investigate how the simulated entities may attain broadly defined goals without detailed step-by-step instructions within a physically precise environment. We describe an experimental test-bed that we have develop and report on some experiments providing quantitative insight into our approach.

2 Simulating Collective Autonomous Behaviour

The actual behaviour of artificial entities is especially important in the context of simulations designed for training personnel or evaluating tactical situations. In such simulations, the behaviour of agents will have an important effect on the final outcome. Unrealistic agent behaviour, e.g., in the form of very limited or even extremely advanced intelligence can result in poor correspondence to real-life situations.

Agent behaviour in a sophisticated simulated environment can be very complex and will involve many collaborating or adversary entities. Intelligence can be employed at very different levels. A simple example will be a team of agents that has to go from one position to another trying to minimise travel time and keep out of trouble. A more complex example of intelligent behaviour can include the decision to cancel the mission of a group of entities and relocating them as a backup for another group. An even more complex situation would involve several adversarial teams, each trying to achieve different goals.

*This research is supported under the Data and Information Fusion DTC by a contract from General Dynamics UK to Imperial College under Project 6.8 (Future Data Fusion Systems Design and Demonstration.

2.1 Related Work

Multi-Agent systems are a very important field in AI since they emerge as a natural way of dealing with problems of distributed nature. Such problems exist in a diversity of areas like military training, games and entertainment industry, management, transportation, information retrieval and many others. The classical approach to AI, until now, has been unable to provide a feasible approach for solving problems of this nature. The need for such tools has lead to the "alternative" approach of behaviour-based systems, popularised by the work of Brooks [7] and Arkin [8]. This approach takes simple behaviour patterns as basic building blocks and tries to implement and understand intelligent behaviour through the construction of artificial life systems. Its inspiration comes from the way intelligent behaviour emerges in natural systems studied by Biology and Sociology. Good discussions on the development of behaviour-based AI can be found in [9, 10] and an extensive treatment of the subject is given in [11].

Multi-agent systems interacting with the real world face some fundamental restrictions. Some of these are:

1. They have to deal with an unknown and dynamic environment
2. Their environment is inherently very complex
3. They have to act within the time frame of the real world
4. The level of their performance should be "acceptable"

In order to meet these requirements, agents have to be able to learn, coordinate and collaborate with each other. Reinforcement Learning emerges as one of the "natural ways" of dealing with the dynamism and uncertainty of the environment. The complexity of the environment and the strict timing constraints however make the learning task extremely difficult. Even simple multi-agent systems consisting of only a few agents within a trivial environment can have prohibitively expensive computational requirements related to learning [12, 13, 14].

The behaviour-based systems have been more successful at dealing with such problems. Biology-inspired models of group behaviour such as Reynold's "boids" [15] and approaches based on potential fields [16] are able to address group behaviour at a reasonable cost. Because of their performance and ability to scale better, they have been widely employed in technology-driven fields such as the computer-games industry [17, 18].

One of the main problems of behaviour-based systems is that their constituents can be very easily caught in local-minima. The question of how to combine different (possibly conflicting) behaviours in order to achieve an emergent intelligence is also very difficult. Multi-Agent Reinforcement Learning in the behaviour domain [19, 20] is an actively explored approach to solve these problems in a robust way.

2.2 Our Proposed Multi-Agent Simulator

Our proposed simulator is designed for behaviourally and visually significant tactical simulations, within a physically accurate setting such as a Terrain Database. The problem we address in this paper is goal-based navigation of a group of autonomous entities in a dangerous terrain. The design of the agent model is based on the assumption that agents will perform "outdoor" missions in a terrain containing obstacles and enemies. It is not very suitable for "indoor" missions like moving inside a building or a labyrinth, where a more specialised approach will be required. A "mission" in our model is defined as the problem of going from some position A to some other position B avoiding being hit by an enemy, and avoiding the natural and artificial obstacles present in the terrain. The success of the mission is measured by the amount of time necessary for the whole group to achieve the goal and the survival rate.

Our approach is based on a hierarchical modular representation of agent behaviour. This method allows for de-coupling the task of group navigation into simpler self-contained sub-problems which are easier to implement in a system having computational constraints due to interaction with real-life entities.

Different decision mechanisms are used to model different aspects of the agent behaviour and a higher level coordination module is combining their output. Such an architecture allows "versatile agent personalities" both in terms of heterogeneity (agent specialisation) within a group and dynamic (i.e. mission-context sensitive) agent behaviour.

The hierarchical modularity of the system also facilitates the assessment of the performance of separate components and related behaviour patterns on the overall success of the mission.

In our current model, we have three basic modules that we call the *navigation* module, *grouping* module and *imitation* module.

- The Navigation Module is responsible for leading a single agent from a source location to a destination location, avoiding danger and obstacles.
- The Grouping Module is responsible for keeping a group of agents together in particular formations throughout the mission.
- The Imitation Module is modelling the case when an inexperienced agent will try to mimic the behaviour of the most successful agents in the group and thus increase its chances of success.

The decisions of these modules are combined at a higher-level module called the *Coordinator Module*. This particular agent model allows modelling of different parts of agent behaviour using different approaches. Some of these approaches may incorporate memory (navigation) while others others can be purely reactive (grouping) and some may depend on the performance of other members in an agent group (imitation and grouping).

2.3 Coordination of Behaviour Modules

The current model of behaviour combination is to get, at each time step, a weighted sum of the separate decisions recommended by each basic module, where the decisions are in the form of a 2D vector representing a request to move in a particular direction with a particular speed:

$$\vec{V}_{overall} = k_{nav} * \vec{V}_{nav} + k_{grp} * \vec{V}_{grp} + k_{imt} * \vec{V}_{imt}$$

The coordinator can for example give priority to the Navigation Module and inhibit the others when it detects that they cause an agent to be trapped in a local minimum. The leader of a group will also honour the Navigation Module, expecting group members to follow him. Another example is when emphasis is given to the Grouping Module, helping a wounded or important agent to stay close to the other members so that it is well protected.

Another degree of freedom comes from the ability of the coordinator to see the "bigger picture" and not only judge how much a module should affect the final outcome, but also give a constructive feedback on how a module should adjust its internal parameters for the good of the mission. The basic decision modules that we consider in this work are described in the following subsections.

2.4 Navigation Module

For the purpose of simplicity and efficiency, the Navigational Module generates moves based on a quantised representation of the simulated environment in the form of a grid. Terrain properties are assumed to be uniform within each grid cell for the purpose of learning and storing information about the terrain. Each cell in the grid represents a position and an "agent action" is defined as the decision to move from a grid cell to one of the eight neighbouring cells. A succession of such actions will result of a completion of a mission. The agents can also access terrain-specific information about features and obstacles of natural (trees, etc.), and artificial origin (buildings, roads, etc.) and also presence of other (possibly hostile) agents. The interaction between an enemy (a hostile agent) and an agent is modelled by an associated risk. This risk is expressed as a probability of being shot (for an agent) at a position, if the position is in the firing range of an enemy. The goal of the agent is to minimise a function G (which in this case is the estimated time of a safe transit to the destination). We use G to define the Reinforcement Learning Reward function as $R \propto 1/G$.

Successive measured values of R are denoted by $R_l, l = 1, 2, \ldots$. These values are used to keep track of a smoothed reward

$$T_l = bT_{l-1} + (1-b)R_l, \quad 0 < b < 1$$

where b is close to 1. A Navigational Module of an agent has a so-called "cognitive map" which is a collection of latest and smoothed rewards for each decision taken at each visited grid cell.

The decision-making element of a Navigation Module is a fully-connected Random Neural Network [1, 2] consisting of 8 neurons (each representing a possible decision). The training is performed by reinforcing the weights of each neuron, depending on the difference between the latest and smoothed rewards; positive difference indicates improvement and negative difference indicates deterioration. The RNN is an analytically tractable spiked neural network model whose mathematical structure is akin to that of queueing networks. It has "product form" just like many useful queueing network models, although it is based on non linear mathematics. The state q_i of the $i-th$ neuron in the network is the probability that it is excited. The q_i, with $1 \leq i \leq n$ satisfy the following system of non linear equations:

$$q_i = \lambda^+(i)/[r(i) + \lambda^-(i)], \tag{1}$$

where

$$\lambda^+(i) = \sum_j q_j w^+_{ji} + \Lambda_i, \quad \lambda^-(i) = \sum_j q_j w^-_{ji} + \lambda_i \tag{2}$$

Here w^+_{ji} is the rate at which neuron j sends "excitation spikes" to neuron i when j is excited, w^-_{ji} is the rate at which neuron j sends "inhibition spikes" to neuron i when j is excited, and $r(i)$ is the total firing rate from the neuron i. For an n neuron network, the network parameters are these n by n "weight matrices" $W^+ = \{w^+(i,j)\}$ and $W^- = \{w^-(i,j)\}$ which need to be "learnt" from input data. Various techniques for learning may be applied to the RNN. These include Hebbian learning (which will not be discussed here since it is too slow and relatively ineffective with small networks), and Reinforcement Learning.

There can be different ways to apply Reinforcement Learning in the RNN model. Given the Goal G that the agent has to achieve as a function to be minimised , we formulate a reward R which is simply $R = G^{-1}$. Let the neurons of the RNN be numbered $1, \ldots, n$. Thus each decision i corresponds to some neuron i. Decisions in this RL algorithm with the RNN are taken by selecting the decision j for which the corresponding neuron is the most excited, i.e., the one with the largest value of q_j. Note that the $l-th$ decision may not contribute directly to the $l-th$ observed reward because of time delays between cause and effect.

Suppose we have now taken the $l-th$ decision which corresponds to neuron j, and that we have measured the $l-th$ reward R_l. Let us denote by r_i the firing rates of the neurons before the update takes place. We first determine whether the most recent value of the reward is larger than the previous "smoothed" value of the reward which we call the threshold T_{l-1}. If that is the case, then we increase very significantly the excitatory weights going into the neuron that was the previous winner (in order to reward it for its new success), and make a small increase of the inhibitory weights leading to other neurons. If the new reward is not better than the previously observed smoothed reward (the threshold), then we simply increase moderately all excitatory weights leading to all neurons, except for the previous winner, and increase significantly the inhibitory weights leading to the previous winning neuron (in order to punish it for not being very successful this time). This is detailed in the algorithm given below. We compute T_{l-1} and then update the network weights as follows for all neurons $i \neq j$:

- If $T_{l-1} \leq R_l$
  - $w^+(i,j) \leftarrow w^+(i,j) + R_l$,
  - $w^-(i,k) \leftarrow w^-(i,k) + \frac{R_l}{n-2},\ if\ k \neq j.$
- Else
  - $w^+(i,k) \leftarrow w^+(i,k) + \frac{R_l}{n-2}, k \neq j,$
  - $w^-(i,j) \leftarrow w^-(i,j) + R_l.$

Since the relative size of the weights of the RNN, rather than the actual values, determine the state of the neural network, we then re-normalise all the weights by carrying out the following operations. First for each i we compute:

$$r_i^* = \sum_1^n [w^+(i,m) + w^-(i,m)], \tag{3}$$

and then re-normalise the weights with:

$$w^+(i,j) \leftarrow w^+(i,j) * \frac{r_i}{r_i^*},$$
$$w^-(i,j) \leftarrow w^-(i,j) * \frac{r_i}{r_i^*}.$$

Finally, the probabilities q_i are computed using the non linear iterations (1), (2), leading to a new decision to move the agent in the direction which corresponds to the neuron which has the largest excitation probability. By using previously acquired information and current sensory input, an agent can start with near-optimal estimates of the rewards and skip an otherwise prohibitively-long learning session and focus on adapting to the dynamic changes in the environment.

2.5 Grouping Module

Grouping behaviour module is based on the idea of social potential fields [16] which is a simple distributed-control approach inspired by the attractive and repulsive forces between charged particle in physics. Although this method has been used in a broader domain (including path-planning), we restrict its usage only to model grouping behaviour for which it is particularly well suited. Using potential fields methods for other purposes like generalised navigation and obstacle avoidance requires dealing with local minima problems and difficult to design force-configurations that can easily nullify the simplicity gained by using the method in the first place.

In our treatment, we restrict the form of the force between agents i and j to:

$$\vec{V}_{i,j} = \left(-\frac{a}{r^\alpha} + \frac{b}{r^\beta} \right) \hat{r}$$

where a, b, α, β are dynamic parameters and the force vector $\vec{V_{i,j}}$ describes the effect of the position of agent j on the decision of agent i. When there is a stable equilibrium point, an entity experiencing such a force will stay at a distance R_0 from the force source, where

$$R_0 = \sqrt[\alpha-\beta]{\frac{b}{a}}$$

The total effect on agent i can be calculated as:

$$\vec{V}_{grp_i} = c * \sum_j \vec{V}_{i,j}$$

By varying the parameters of each force, different behaviours like attraction to an agent, repulsion from an agent or trying to stay within some distance from an agent can be modelled - the last being especially important in forming spatially localised groups.

These behaviours are very similar and can be used to get the effect of the *collision avoidance* and *flock centring* rules as described by Reynolds [15].

By setting up a two-way mesh of forces between a number of agents, for example, a spatially localised group can be created that will try to stay together.

Another simple example is a one-way mesh of forces from the leader of the group to the other members, suggesting that they should stay close and follow if necessary the leader, without having any effect on his decision making.

2.6 Imitation Module

The imitation module proposes a decision which is a weighted sum of the navigational decisions of some of the members of the agent group:

$$\vec{V}_{imt_i} = \sum_{j \in S} w_j * \vec{V}_{nav_j}$$

The weight distribution can be dynamic, in order to reflect the group members which are currently observable or known to be experienced, for example. The purpose of imitation is to efficiently take advantage of experience without going through the trouble of actually acquiring it - that is, it has a much lower computational cost, compared to the other methods.

The *velocity matching* flock behaviour described in the work of Reynolds [15] which he defines as "attempt to match velocity with nearby flocks" is a very similar idea.

3 Experiments

We have developed a multi-agent simulator for testing our ideas and measuring performance of agent behaviour. The current simulation testbed was used to perform a series of experiments under different behaviour module configurations. Figure 1 is an example of the current simulator in operation.

The terrain size is 200 by 200 units and it is overlaid by a 50x50 grid. There are 300 trees and 10 buildings present. A group of 8 agents are in the process of moving from the lower-left corner of the terrain to the destination area designated by small flags in the upper right part of the terrain. The agent group consists of one leader and seven group members. The distinction between the leader and the rest of the agents is dictated by the way social potential field (SPF) forces are configured.

There are two types of SPF forces involved in this setup:

- A two-way force (F_1) between group members (excluding the leader). The force parameters are ($a = 1, \alpha = 1.6, b = 16, \beta = 3.6$).

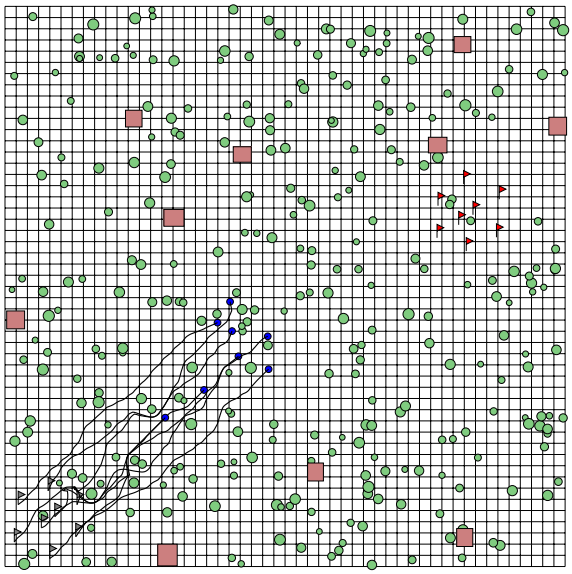

Fig. 1: An example of an ongoing mission in the agent simulator

- A one-way force (F_2) from the leader to the group members. The force parameters are ($a = 1, \alpha = 1.6, b = 4, \beta = 3.6$).

The parameters have the effect of keeping an inter-group distance of approximately 4 units, and distance between group members and the leader of approximately 2 units - in this way, the group is surrounding the leader. The leader itself is not affected in any way by the other agents.

It is very important that we are able to evaluate and compare the performance of different modes of behaviour both during and after simulation runs. The performance metrics used in the simulations presented in this paper are defined below:

- **Group Tension** is defined as the average magnitude of the effective SPF force experienced by agents in a group. By an effective SPF force we mean the vector summation of all SPF forces acting on a particular agent (which happens to be the output of the Grouping module for that agent). A small value for the group tension should indicate that the group is well-formed, while a greater value should indicate internal stress within the group related to bad spatial formation. This metric is a particularly good indicator of congestion within a group. However, the group tension will not identify cases in which an agent is separated from the group, since SPF forces decay quickly with distance.

- **Group Radius** is defined as the average distance from an agent to the geometric centre of its group. It is an indicator of how good the spatial formation of a group of agents is. An increase in the group radius, for example, can be used to detect cases when an agent has lost proximity with its group or when a group is breaking apart.

- **Travel Distance** is defined as the average distance travelled by the agents in a group. It is a good estimate of the mobility characteristics of a group as a whole.

- **Travel Energy** is defined as the average energy spent by members of an agent group. Even very similar mobile agents can exhibit significant differences in energy consumption based on specific environmental conditions or agent specifics (i.e. vehicle wear), therefore it is pretty much impossible to devise an accurate energy metric for a generic mobile agent. However, it is still possible to to devise a sensible measure of energy consumption for a range of mobile agents by using simple physics principles. We assume that increasing the speed of an agent costs energy while breaking is almost free (as in most ground vehicles). Therefore, we define travel energy for an agent as the sum of the positive kinetic energy increments over the time of the mission.

To illustrate the performance of different behaviour modes in the agent simulator, we show example measurements of group tension, group radius, travel distance and energy for a bigger simulated environment. The environment is a 2000x2000 terrain overlaid by a 200x200 grid. There are 6000 trees and 100 buildings. The same agent configuration (8 agents - 1 leader and 7 group members) is used. Random missions (random initial position and random destination within the same terrain) are generated and an average value for the specified performance metrics over all the missions is calculated. Figure 2 show measurements for 1000 different mission simulations. Another set of measurements with the addition of 5 uniformly distributed static enemies with firing ranges covering a small portion of the terrain is shown in Figure 3.

There are four different behaviour modes used. The acronyms used in the figures have the following meanings:

- **RL -** Everybody uses Navigation Module

- **RL+IMI -** Everybody uses Navigation Module, group members (i.e. except the leader) also use Imitation (i.e. group members imitate the leader)

- **RL+SPF -** Everybody uses Navigation and Grouping Modules

- **RL+SPF+IMI -** Everybody uses Navigation and Grouping Modules, group members also use Imitation

As far as group tension is considered, the best performance is achieved by RL+SPF followed by RL+SPF+IMI. The other two modes of operation give worse results, which is understandable because they do not employ social potential fields. On the group radius metric, RL+SPF and RL+SPF+IMI show approximately the same performance, with the RL+SPF+IMI mode being slightly better (the difference is more pronounced when enemies are introduced in the terrain). On the travel distance metric, we have almost identical performance for all methods except for RL+SPF, which is a bit worse than the others (it travels

slower). The best performer on the travel energy metric is RL+IMI followed closely by RL and RL+SPF+IMI. The most inefficient method is RL+SPF with twice the travel energy of the other methods. We can say that the RL+SPF+IMI combination, therefore, offers a reasonable performance trade-off between different behaviours. Of course, these simulations were performed with static agent personalities where the agent behaviour combination was fixed during the simulation runs. A dynamic behaviour, based on situational awareness will certainly outperform these static behaviour modes.

4 Conclusions

Modern tactical simulators often require the representation of complex autonomous behaviours within a realistic setting. The idea is to ask questions about "what would happen if ..." for a *team* of agents, in the context of a real environment and potential events. This challenge is the focus of the work addressed in this paper where we consider how broadly defined goals can be used by teams of simulated autonomous entities to achieve the goals. We consider a combination of individual and social control schemes as a way of representing complex behaviours without providing precise prescriptive directions.

We have discussed the conceptual issues which arise in this key area of simulation, and presented some design principles and a practical implementation. We propose a novel approach to automatically control the motion of synthetic agents in pursuit of broad goals, by combining reinforcement learning, social potential fields and imitation. Experiments show that our modular behaviour-based approach is able to combine simple behaviour modules such that the emergent composite behaviour outperforms each of its constituents.

Future work will discuss how the teams' own observed behaviour can be used to adaptively improve future behaviour during the same mission. This will include changing the manner in which different control modules are combined. We will also investigate the role of diversity, as a means to achieve overall better team performance. In the context of adversarial teams, we will study how one can infer the goal pursued by a team so as to provide better tracking and countermeasures.

References

[1] Erol Gelenbe. Random neural networks with positive and negative signals and product form solution. *Neural Computation*, **1** (4), pp 502-510, 1989.

[2] Erol Gelenbe. Learning in the recurrent random neural network. *Neural Computation*. **4**, pp 154-164, 1993.

[3] Erol Gelenbe. Modeling CGF with learning stochastic finite-state machines. *Proc. 8-th Conference on Computer Generated Forces*, pp. 113–116, Orlando, 1999.

[4] Erol Gelenbe, Esin Şeref, and Zhiguang Xu. Discrete event simulation using goal oriented learning agents. *AI, Simulation & Planning in High Autonomy Systems*, SCS, Tucson, Arizona, 2000.

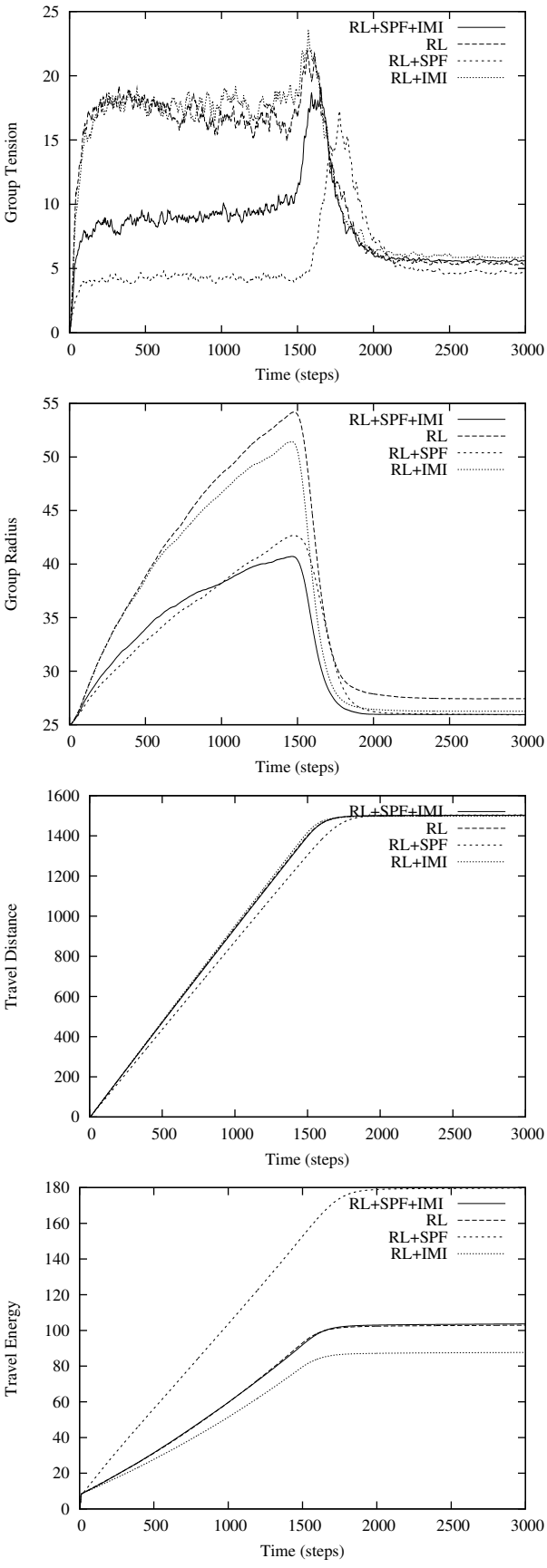

Fig. 2: Average performance metrics for 1000 simulations over a terrain without enemies

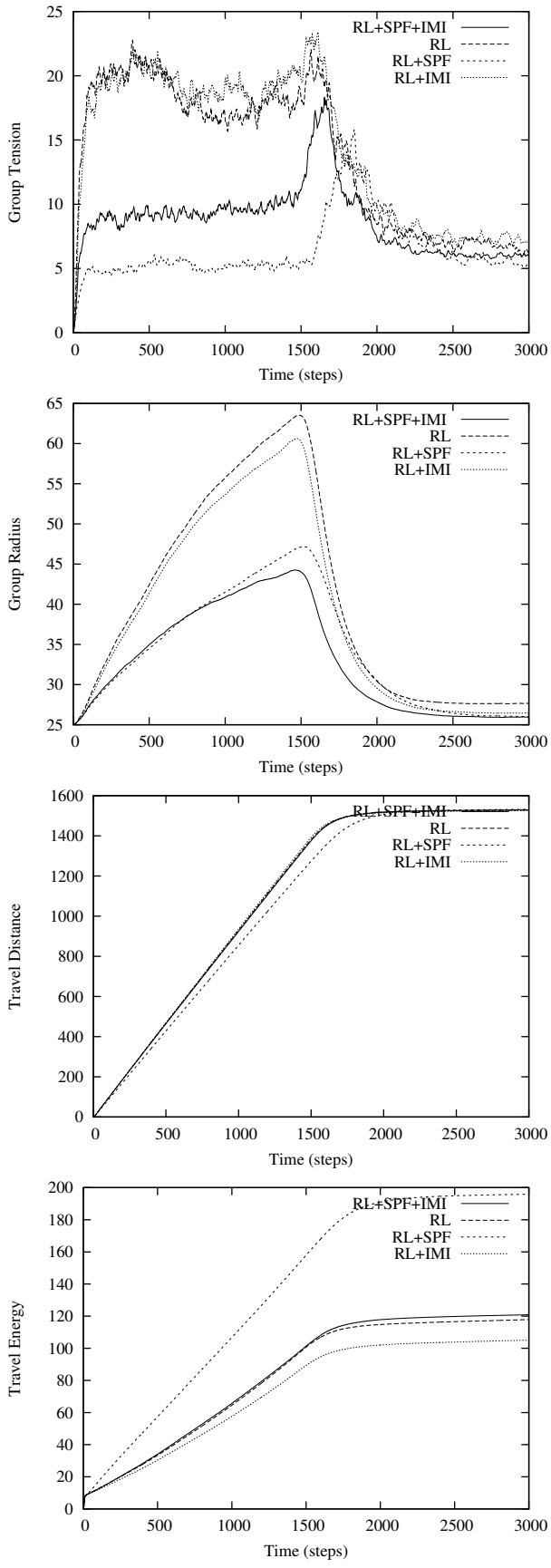

Fig. 3: Average performance metrics for 1000 simulations over a terrain with static enemies

[5] Erol Gelenbe, Esin Şeref, and Zhiguang Xu. Simulation with Learning Agents. *Proceedings of IEEE*, **89**(2):148–157, 2 2001.

[6] Billy Foss, Erol Gelenbe, Khaled Hussain, Niels D. Lobo, and H. Bahr. Simulation driven virtual objects in real scenes. *Proc. ITSEC 2000*, Orlando, 2000.

[7] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, **RA-2**(1):14–23, 1986.

[8] Ronald C. Arkin. Motor schema-based mobile robot navigation. *International Journal of Robotics Research*, **8**(4):92–112, 1989.

[9] Luc Steels. The Artificial Life Roots of Artificial Intelligence. *Artificial Life*, **1**:75–110, 1993.

[10] Rodney A. Brooks. *Cambrian Intelligence: The Early History of The New AI*. The MIT Press, Cambridge, Massachusetts, 1999.

[11] Ronald C. Arkin. *Behavior-Based Robotics*. The MIT Press, Cambridge, Massachusetts, 1998.

[12] Ming Tan. Multi-Agent Reinforcement Learning: Independent versus Cooperative Agents. In *International Conference on Machine Learning*, pp. 330–337, 1993.

[13] Norihiko Ono and Kenji Fukumoto. Multi-agent reinforcement learning: A modular approach. In *Proc. of 2nd International Conference on Multi-Agent Systems*, pp. 252–258. AAAI Press, 1996.

[14] Norihiko Ono and Kenji Fukumoto. A modular approach to multi-agent reinforcement learning. In Gerhard Weiss, editor, *Distributed Artificial Intelligence Meets Machine Learning*, p. 167. Springer-Verlag, 1997.

[15] Craig W. Reynolds. Flocks, Herds, and Schools: A Distributed behavioural Model. *Computer Graphics*, **21**(4):25–34, 1987.

[16] John H. Reif and Hongyan Wang. Social Potential Fields: A Distributed behavioral Control for Autonomous Robots. In A. K. Peters, editor, *International Workshop on Algorithmic Foundations of Robotics (WAFR)*, pp. 431–459, Wellesley, Massachusetts, 1998.

[17] Craig W. Reynolds. Steering behaviors for Autonomous Characters. In *Game Developer Conference*, 1999.

[18] Dave C. Pottinger. Implementing Coordinated Movement. *Game Developer Magazine*, January 1999.

[19] Maja J. Mataric. Reinforcement learning in the multi-robot domain. *Autonomous Robots*, **4**(1):73–83, 1997.

[20] Tucker Balch. *Behavioral Diversity in Learning Robot Teams*. PhD thesis, Georgia Institute of Technology, 1998.