Bayesian Imitation of Human Behavior in Interactive Computer Games

Bernard Gorman[1], Christian Thurau[2], Christian Bauckhage[3] and Mark Humphrys[1]
[1]Dublin City University bgorman@computing.dcu.ie
[2]Bielefeld University cthurau@techfak.uni-bielefeld.de
[3]Deutsche Telekom Laboratories christian.bauckhage@telekom.de

Abstract

Modern interactive computer games provide the ability to objectively record complex human behavior, offering a variety of interesting challenges to the pattern-recognition community. Such recordings often represent a multiplexing of long-term strategy, mid-term tactics and short-term reactions, in addition to the more low-level details of the player's movements. In this paper, we describe our work in the field of imitation learning; more specifically, we present a mature, Bayesian-based approach to the extraction of both the strategic behavior and movement patterns of a human player, and their use in realizing a cloned artificial agent. We then describe a set of experiments demonstrating the effectiveness of our model.

1 Motivation and Related Work

Interactive computer games, with their increasingly complex virtual worlds and ability to record the actions of humans within them, present interesting opportunities to the pattern-recognition community [7]. So far, the AI systems employed in computer games have typically eschewed pattern-recognition and machine learning, favoring outmoded finite-state machines and graph-searching algorithms [3, 8]. The idea of imitation learning –already prominent in robotics [10]– has received scant attention, despite its obvious suitability; substantial libraries of samples are available online, new samples can be generated in large quantities and with ease, and these samples encode nothing less than the output of competitive human reasoning tasks. The objective of imitation learning in computer games, then, is to deduce the patterns within these recorded gameplay sessions, and 'reverse-engineer' the player's decision-making process from its observed results.

Several investigations [6, 14] have identified long-term planning as particularly crucial in determining the degree to which an agent is perceived as 'humanlike'. In order to learn long-term strategic behaviors from human demonstration, we develop a model designed to emulate the notion of program level imitation [2] – that is, to identify the demonstrator's intent, rather than simply reproducing his actions. Because modern games typically require players to simultaneously engage in a variety of short, medium and long-term tasks, this requires an approach capable of recognizing and filtering the goal-oriented behavior patterns from the gameplay sample, discarding contemporaneous actions which do not contribute towards reaching the strategic objectives.

In the course of a game session, human players also exhibit actions other than simply moving along the environment surface, including jumps, weapon changes and discharges, crouches, etc. Often, players can only attain certain goals by performing one or more such actions; they therefore have an important functional element. Concerning the believability of agents, it is also vital to reproduce the aesthetic qualities of these actions - it is the characteristic smoothness of human motion, as contrasted against the robotic movement of traditional artificial agents, that distinguishes live players from AI. For the purposes of our agent, then, a means of imitating such actions is essential.

The first-person shooter (FPS) genre, where players explore 3D environments littered with weapons, bonus items, traps and pitfalls, and defeat as many opponents as possible, was chosen for our work because it provides a relatively direct mapping of human decisions onto agent actions. In other genres such as sports or strategy games, significant swathes of gameplay are dictated by the software's built-in AI; in FPS games, by contrast, the player assesses and responds to his situation, and his decisions are translated into observable in-game actions. Due to its prominence [6, 7], we use iD Software's QUAKE II® as our testbed. In order to extract the required data from its recorded DM2 or demo file format –consisting of the network traffic received during the game – and to realize the in-game agents, we employ our own API [1].

In previous work, Thurau et al. [11] proposed a hierarchical representation of the differing behaviors encoded in QUAKE II® demos ranging from long-term strategies to short-term reactions, and outlined an approach to movement imitation based on the theory of action primitives [12]. Gorman and Humphrys [5] subsequently introduced a method

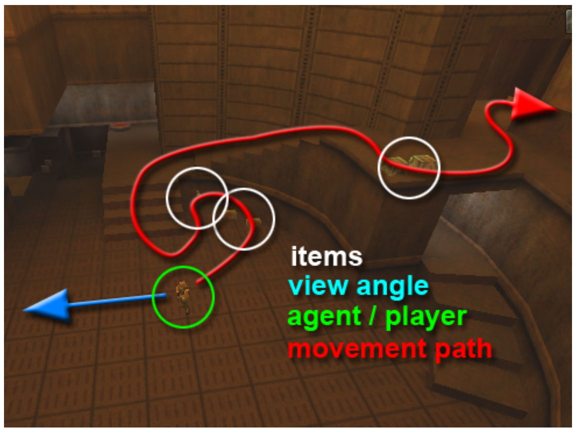

Figure 1. Q2 environment. The player strategically cycles the 3D map, collecting items to maximize his advantage. The avatar's movement direction is independent of his viewing angle, giving rise to the characteristically human smoothness of the movement path.

of reproducing the strategic navigational behaviors of a human player, and described initial attempts to integrate this with the action primitives framework. Simultaneoulsy, Thurau et al. [13] extended their framework using a Bayesian model of imitation in infants developed by Rao et al. [9].

In this paper, we propose an integration of both these models to create a unified imitation mechanism and present experiments which demonstrate its efficacy.

2 Learning Strategic Patterns

In QUAKE II$^{®}$, experienced players traverse the environment methodically, controlling important areas and collecting items to strengthen their character. We therefore define the player's strategic goals to be the items scattered at fixed points around each level. By learning the mappings between the player's status and his subsequent item pickups, the agent can adopt observed strategies when appropriate, and adapt to situations which the player did not face.

We read the set of player locations $\vec{l} = [x, y, z]$ from the recording, and cluster them to produce a reduced set of typical positions (nodes). We also construct a matrix of edges E, where $E_{ij} = 1$ if the player was observed to move from node i to node j. This topological map of the environment can now be viewed as a Markov Decision Process, with the nodes corresponding to states and the edges to transitions.

The player's inventory –the list of what quantities of which items he currently possesses– is also read at each timestep. The inventory vectors represent varying situations faced by the player during the game. We can now construct a set of paths which the player followed through the environment while in each inventory state. These paths consist of a series $\{c_{i,1}, c_{i,2}, \ldots, c_{i,k}\}$ of transitions between clusters where $c_{i,j}$ is a single node along a sequence. Each path begins at the point where the player enters a given state, and ends where he exits that state. In other words, collecting an item causes the player's inventory to shift towards a different prototype.

Having obtained the different paths pursued by the player in response to his changing condition, we turn to reinforcement learning to learn his behavior.

A modified version of the value iteration algorithm is employed to compute the utility values for every node under each inventory state prototype. We consider an action a to be the choice to move to a given node from the current position, and assign rewards such that the agent is guided along the same paths as the human. The transition probabilities and rewards are therefore given as

$$P(c' = j | c = i, a = j) = E_{ij} \text{ and } R(p_i, c_{i,j}) = j \qquad (1)$$

That is, each successive node along the path's length receives a reward greater than the last, until the last cluster (at which an inventory state change occurred) is assigned the highest reward. If a path loops back or crosses over itself, the higher values will overwrite the previous rewards, ensuring that the agent will be guided towards the terminal node while filtering out any non-goal-oriented diversions. Thus, the agent will emulate the player's program-level behavior, instead of simply duplicating his movements.

In situations where several items are of strategic benefit, the player will intuitively weigh their relative importance before deciding on his next move. To model this, we adopt a fuzzy clustering approach, expressing the agent's current inventory as a membership distribution across all prototype inventory states. Another important factor is the human's understanding of object transience: a collected item will be unavailable until the game regenerates it after a fixed interval. To capture this, we introduce an activation variable in the computation of the membership values:

$$m_p(\vec{s}) = \frac{a(o_p) d^{-1}(\vec{s}, \vec{p})}{\sum a(o_i) d^{-1}(\vec{s}, \vec{i})} \qquad (2)$$

where $\vec{s}$ is the current inventory state, $\vec{p}$ is a prototype inventory state, $a = 1$ if the object o at the terminal node of the path associated with the given prototype is present and $a = 0$ otherwise, and d^{-1} is a proximity function. This implicitly defines the agent's current goals, which (as will be shown later) facilitates integration with the Bayesian motion-modeling system. The utility configurations associated with each prototype are then weighted according to the membership distribution, and the adjusted configurations superimposed; we also apply an online discount to prevent backtracking. The final utilities are thus computed as

$$U(c) = \gamma^{e(c)} \sum V_p(c) m_p(\vec{s}) \tag{3}$$

$$c_{t+1} = \max_y U(y), \quad y \in \{x | E_{c,x} = 1\}$$

where $U(c)$ is the final utility of node c, γ is the discount, $e(c)$ is the number of times the player has entered cluster c since the last state transition, and $V_p(c)$ is the original value of node c in state prototype p.

3 Bayesian Action Primitives

It is not sufficient to simply identify the player's goals and how (s)he reached them; it is also necessary to capture the actions executed in pursuit of these goals. In [13], Thurau et al. describe an approach based on Rao's Bayesian inverse-model for action selection in infants and robots [9]. The choice of action at each timestep is expressed as a probability function of the subject's current state c_t, next state c_{t+1} and goal state c_g, as follows

$$P(a_t | c_t, c_{t+1}, c_g) = \frac{P(c_{t+1} | c_t, a_t) P(a_t | c_t, c_g)}{\sum_u P(c_{t+1} | c_t, a_u) P(a_u | c_t, c_g)} \tag{4}$$

This model fits into the strategic navigation system almost perfectly; the states (position clusters) c_t and c_{t+1} are chosen according to utility values, while the current goal is given by the membership distribution. To derive the probabilities, we read the actions taken by the player as a sequence of vectors

$$\vec{v} = [\Delta\text{yaw}, \Delta\text{pitch}, \text{jump}, \text{weapon}, \text{firing}]$$

Similar to [4], clustering these action vectors yields a set of action primitives, each of which amalgamates a number of similar actions into a single unit of behavior. The priors can now be derived by direct examination of the observation data.

Several important adaptations must be made in order to use this model in the game environment. Firstly, Rao's model assumes instantaneous transitions between states, whereas in QUAKE II®, multiple actions may be performed while moving between successive clusters. We therefore express $P(c_{t+1} = c_j | c_t = c_i, a_t = a_i)$ as a soft-distribution of all observed actions on edge $E_{c_t, c_{t+1}}$. Secondly, Rao assumes a single unambiguous goal, whereas we deal with multiple weighted goals in parallel. We thus perform a similar weighting of the probabilities across all active goal clusters. Finally, Rao's model assumes that each action is independent of the previous one. In QUAKE II®, however, an action is constrained by that of preceding timestep. We therefore introduce an additional dependency

Cluster density	20%	30%	50%
Position MAE (Units)	21.32	19.22	**17.76**
Position MAPE (%)	2.69	2.42	**2.32**
Yaw RMSE (°)	4.78	3.84	**2.71**
Pitch RMSE (°)	0.30	0.17	**0.11**

Table 1. Variation of mean error with density

in our calculations such that the final probabilities are

$$\sum_g m_g P(a_t | c_t, c_{t+1}, c_g) \frac{P(a_t | a_{t-1})}{\sum_u P(a_u | a_{t-1})}$$

4 Experiments and Discussion

To test the model, we utilized 35 gameplay samples of varying length and complexity, spread across three distinct environments. In each case, the agent attempted to reproduce the human's behavior with cluster densities of 20%, 30% and 50% of the total number of samples, both for constructing the topological map and for deriving the action primitives. Because the agent's task was not to simply copy the human, but rather to deduce his objectives and pursue them independently, a frame-by-frame comparison of the artificial player's actions against the demonstrator's is not appropriate. Instead, we compute the error between the agent's actions while traversing each edge and those taken by the human on the same edge while in the same goal state. See Table 1 for average error rates with varying cluster densities.

As can be seen in the charts below, the model resulted in a highly accurate derivation of the human player's behavior patterns. The MAP error between the agent's position and that of the human did not exceed 7.5%, and was generally much lower; the highest recorded RMS error in the agent's yaw was $\approx 10°$ at the lowest density, while its pitch stayed within $1.36°$ of the observed target. This low pitch error reflects the fact that the agent was engaged in imitating the player's strategic navigational behaviors. Variations in the horizontal plane were therefore greater (moving around corners, etc.) than those in its vertical, which would be more pronounced in a combat scenario. The numerical results correlated with the very humanlike visual in-game appearance of the imitation agents.

While the choice of cluster numbers revealed itself to be dependent upon the complexity of the environment and the size of the gameplay sample, the charts demonstrate a trend towards greater accuracy with more clusters (see Table 1). In certain cases, however, this did not hold true. We found this to be caused by the overcrowding of the topological map with waypoint nodes, which forced the agent to make more course corrections than were necessary for accu-

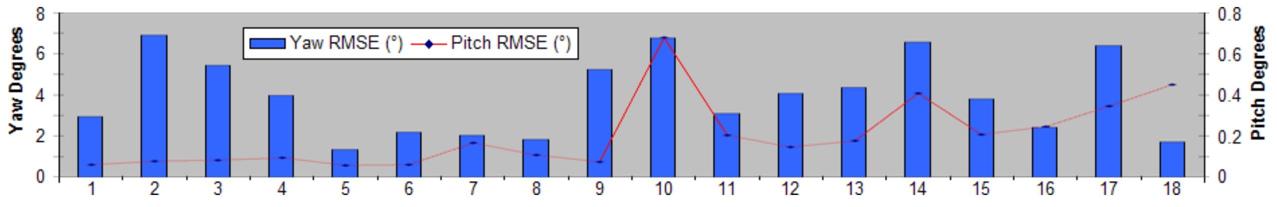

Figure 2. Average Yaw/Pitch RMSE for selected samples

Sample	Position MAPE %	Yaw RMSE (°)	Pitch RMSE (°)
1	0.71	1.52	0.10
5	0.91	0.20	0.16
10	0.98	1.63	0.03
15	2.88	2.29	0.07
20	2.23	7.23	0.09
25	3.21	4.85	0.09
30	5.83	2.06	0.13
35	2.11	0.71	0.09

Table 2. Error rates for 50% cluster density

rate reproduction of the player's navigation. This, in turn, often had a negative impact on the accuracy of the action primitives. Higher topological densities are also undesirable from a technical point of view, since they increase the duration of the learning process. Given the relatively small benefits of increasing the topological density versus the significant benefits of increasing the number of primitives (Table 1), a useful guideline is to use the minimum number of nodes required to yield a sufficiently detailed representation of the environment, while maximizing the action primitives according to any relevant resource considerations. In general, we found that cluster densities of less than 20% were not sufficient to capture the topology of the environment, while densities of greater than 50% often resulted in a decrease in performance, both numerically and perceptually.

5 Conclusion

In this paper, we proposed modern computer games as an ideal domain for pattern recognition research, given the quantity of available samples, the complexity of the data they contain, and the ease of generating new samples. We then presented a reinforcement-learning approach to deriving the strategic behaviour patterns of a human player from a recorded game session, and described how we combined this with a Bayesian framework for imitating the player's individual actions. Tests of agents trained using the unified model showed it to be extremely effective in learning and reproducing the goal-driven behaviour of the demonstrator, while conveying the strong impression of being a human rather than artificial player.

References

[1] C. Fairclough, M. Fagan, B. MacNamee, and P. Cunningham. Research Directions for AI in Computer Games. Technical report, Trinity College Dublin, 2001.

[2] A. Fod, M. Matarić, and O. Jenkins. Automated derivation of primitives for movement classification. *Autonomous Robots*, 12(1):39–54, 2002.

[3] B. Gorman, M. Fredriksson, and M. Humphrys. QASE – an integrated api for imitation and general ai research in commercial computer games. In *Proc. CGAMES Int. Conf. Computer Games*, pages 207–214, 2005.

[4] B. Gorman and M. Humphrys. Towards integration of strategic planning and motion modelling in interacctive cojmputer games. In *Proc. Int. Conf. Computer Game Design & Technology*, pages 92–99, 2005.

[5] J. E. Laird. Using a Computer Game to develop advanced AI. *IEEE Computer*, pages 70–75, July 2001.

[6] J. E. Laird and M. v. Lent. Interactice Computer Games: Human-Level AI's Killer Application. In *Proc. AAAI*, pages 1171–1178, 2000.

[7] A. Naraeyek. Computer games – boon or bane for ai research. *Künstliche Intelligenz*, pages 43–44, February 2004.

[8] R. Rao, A. Shon, and A. Meltzoff. A Bayesian Model of Imitation in Infants and Robots. In K. Dautenhahn and C. Nehaniv, editors, *Imitation and Social Learning in Robots, Humans, and Animals: Behavioural, Social and Communicative Dimensions*. Cambridge University Press,, 2004.

[9] S. Schaal. Is Imitation Learning the Route to Humanoid Robots? *Trends in Cognitive Sciences*, 3(6):233–242, 1999.

[10] C. Thurau, C. Bauckhage, and G. Sagerer. Learning Human-Like Movement Behavior for Computer Games. In *Proc. Int. Conf. on the Simulation of Adaptive Behavior*, pages 315–323. MIT Press, 2004.

[11] C. Thurau, C. Bauckhage, and G. Sagerer. Synthesizing Movements for Computer Game Characters. In *Pattern Recognition*, volume 3175 of *LNCS*, pages 179–186. Springer, 2004.

[12] C. Thurau, T. Paczian, and C. Bauckhage. Is Bayesian Imitation Learning the Route to Believable Gamebots? In *Proc. GAME-ON North America*, pages 3–9, 2005.

[13] S. Wallace and J. Laird. Behavior bounding: Toward effective comparisons of agents & humans. In *Proc. IJCAI*, pages 727–732, 2003.