Imitation learning at all levels of game-AI

Conference Paper · January 2004

CITATIONS
49

READS
2,942

3 authors, including:

Christian Thurau
Game Analytics
67 PUBLICATIONS **2,739** CITATIONS
SEE PROFILE

Christian Bauckhage
University of Bonn
456 PUBLICATIONS **7,717** CITATIONS
SEE PROFILE

Imitation learning at all levels of Game-AI

Christian Thurau, Gerhard Sagerer
Applied Computer Science
Bielefeld University
P.O. Box 100131, 33501 Bielefeld, Germany
{cthurau,sagerer}@techfak.uni-bielefeld.de

Christian Bauckhage
Centre for Vision Research
York University
4700 Keele Street, Toronto M3J 1P3, Canada
bauckhag@cs.yorku.ca

ABSTRACT

Imitation is a powerful mechanism the human brain applies to extend its repertoire of solutions and behaviors suitable to solve problems of various kinds. From an abstract point of view, the major advantage of this strategy is that it reduces the search space of apropriate solutions. In this contribution, we discuss if and how the principle of imitation learning can facilitate the programing of life-like computer game charecters. We present different algorithms that learn from human generated training data and we show that machine learning can be applied on different levels of cognitive abstraction.

Introduction

Considering the past two decades, we can clearly observe a coevolution of commercial computer games, computer graphics and networking. However, for the time being, this dynamic between game industry and academic research seems rather an exception than the rule. Although the potential for another coevolution is obvious, behaviour programming for game characters and artificial intelligence (AI) or machine learning (ML) research hardly inspired each other.

Of course, ideas from academia have entered game AI programming. But it is striking, though, that the two most prevalent AI techniques in game programming, i.e. the A* algorithm and finite state machines, are somewhat old-fashioned. A*-search for pathfinding was introduced more than three decades ago (Hart et al. 1968) and finite state machines with outputs even date back to the 1950s (Mealy 1955, Moore 1956)

The main reason why more recent results just merely influence game character programming is that algorithms that would produce the versatility and flexibility of human players are still not available. So far, research on autonomous agents mainly focused on robots that navigate through the physical world. And even though roboticists recognise the importance of *learning from demonstration* in order to constrain the search space spanned by this task (Schaal 1999), it's fair to say that uncontrollable environmental dynamics and sensor noise still consume more intellectual efforts than techniques for behaviour representation and learning. However, this situation is about to change as the AI and ML communities are beginning to discover the merits of computer games (Amir & Doyle 2002, Laird 2001, Le Hy et al. 2004, Nareyek 2004, Sklar et al. 1999, Yannakakis & Hallam 2004, Spronck et al. 2002)

In this paper, we will discuss the possible impact of *imitation learning* techniques for computer games. To this end, we will briefly summerise behavioural, neurobiological and AI and ML perspectives on imitation learning. Then we will identify different levels of human behaviour that occur in computer games and consequently will require algorithmic solutions. Following an idea discussed in (Bauckhage et al. 2003), we will thus report on different techniques of analysing the network traffic of multiplayer games in order to realize game agents that exert human-like behaviour learned from examples.

Imitation Learning for Games

Numerous behavioural science experiments document that infants endeavour to produce a behaviour previously demonstrated to them. Some psychological studies on imitation learning even suggest that infants devote most of their time to the imitation of observed behaviours (Rao & Meltzoff 2003).

Obviously, imitation requires a mechanism to map percepts onto actions. And indeed, neurophysiological examinations indicate that there are particular brain areas specialised in imitation (Hietanen & Perrett 1996). After finding neurons that were specific to the execution of goal related limb movements, a connection to imitation came with the discovery of mirror neurons (at least in the brain of macaque monkeys) which are active during the observation as well as the execution of a task (Kohler et al. 2002).

Backed by these results from behavioural science and neurobiology it is no surprise that the idea of imitation learning is getting ever more popular in AI and robotics. Current robotics research on imitation learning mainly concentrates on sub-symbolic techniques like neural networks and fuzzy control. However, it is well known that such low-level approaches do not scale well to situations of many degrees of freedom. As a consequence, the concept of movement primitives was introduced to encode complete temporal behaviours (Fod et al. 2002, Schaal et al. 2003). In fact, there

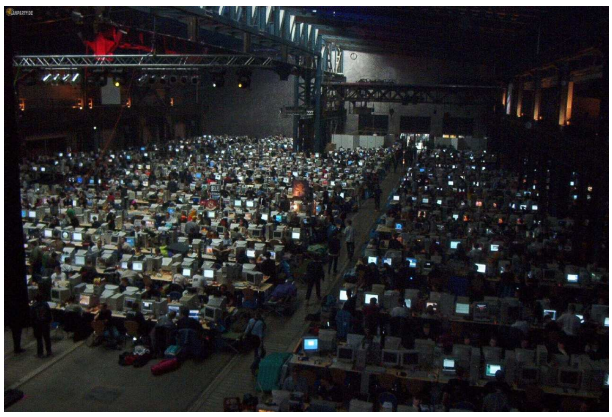

Figure 1: Training sample generation at a “lan-party”

is evidence that the human brain uses movement primitives to produce goal oriented actions since using such primitives considerably reduces the number of parameters that must be learned (Thoroughman & Shadmehr 2000). Movement primitives are thus closely related to mirror neurons some of which are believed to be high-level representations of behaviour.

The capabilities of the human brain of course extend to virtual worlds and learning how to play a computer game is a process of training and imitating experienced players. In the following, we will thus argue that –similar to robotics– learning from demonstration can provide an avenue to behaviour programming for computer game characters. Unlike present day robotics, however, computer games provide an excellent testbed for the learning of complex behaviours. For most genres, the behaviour of game characters will be composed of *reactive*, *tactical* and *strategic* decisions. While the latter, higher cognitive aspects are still widely neglected in robotics (since the problem of sensor noise is so predominant), computer games allow to study them rather effortlessly.

To substantiate our arguments, we will base the discussion on the example of ID software’s game QUAKE II®. First, we shall describe what kind of *percepts* this game provides as input for imitation learning. Then, we will present algorithmic solutions for different kinds of behaviour.

Why Quake ?

QUAKE II® is a so called *Ego* or *First-Person-Shooter*. Figure 2 shows a typical game situation the way a human player views the gaming world, from a first-person perspective.

The main goal of the game is to get the most points in a fixed time span. Points are gained by shooting enemy players (so far we are not using team based game modifications, therefore all other players are opponents).

The game takes place in a 3D gameworld which is loosely based on the real world. The human players can move around freely, their actions are bound to the game physics, which are also based on the real world, not all places are always reachable.

Different types of items (weapons, armor, health packages) are distributed at fixed positions around the map. If an item is picked up, it reappears about thirty seconds later at the same position. From these item positions arises one strategical component of the game. Winning is a lot easier by smart item control, get the best items for yourself and only allow the most weak weapons and armors to be picked up by your enemies. But there are of course endless possibilities of different strategies, which all might lead to a successful gameplay.

Winning a game does not only depends on good aiming, smart playing is often a better way to success - maybe that is the reason why even game bots playing with superhuman aiming are still beaten by good, experienced human players [1].

So why would we want to use such a complex game for imitating human controlled game agents? Simpler games, more focused onto one aspect of human acting could maybe provide better results with less effort, besides, by using a homemade game we could avoid the troubles of interacting with someone else’s (mostly closed source) software.

There are basically two answers to that question. First, imitating humans wandering around a 3D virtual world and performing tasks of different complexity is not only of interest for the gaming community and might provide further insights on the modeling of human behaviours in general. The level of abstraction compared to the real world is a lot lower than in Chess, Tetris or PacMan. In addition, unlike in the real world, we have perfect sensor data. The second point is, that by using a commercially successful game, we have access to an incredible huge database of records of humans playing the game – these so called demo files can be recorded by ourselves (Figure 1 shows a possible location for demo recording) or just downloaded from numerous sites on the internet. Having an almost unlimited amount of training samples, showing nothing less than humans performing complex tasks in complex environments is a unique (so far almost unrecognized) setting.

The training data or demo files we are dealing with are records of the network traffic. They contain information about the exact locations (x, y, z) the player assumed, nearby items and other players. Temporary entities like sounds, and flying projectiles are also included. There is no need for a visual analysis of a game scene, since all necessary information is already available on a cognitive higher level. The same applies to the player actions, they are included as simple velocity and position vectors.

Categories of Behavior

Our current model for the imitation of a human player’s behaviour in QUAKE II® consists of three separate layers as

[1] `http://botchallenge.com` offers a competition on who is the fastest human player in beating *nightmare* (which is somehow equal to superhuman) skilled game-agents in the game QUAKE III®

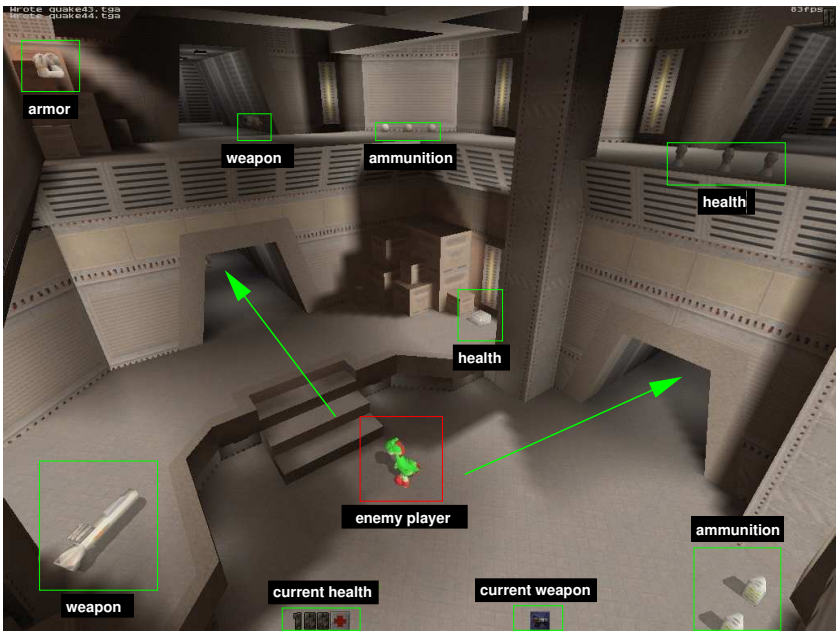

Figure 2: A typical situation in the FPS QUAKE II® , augmented by some entity descriptions.

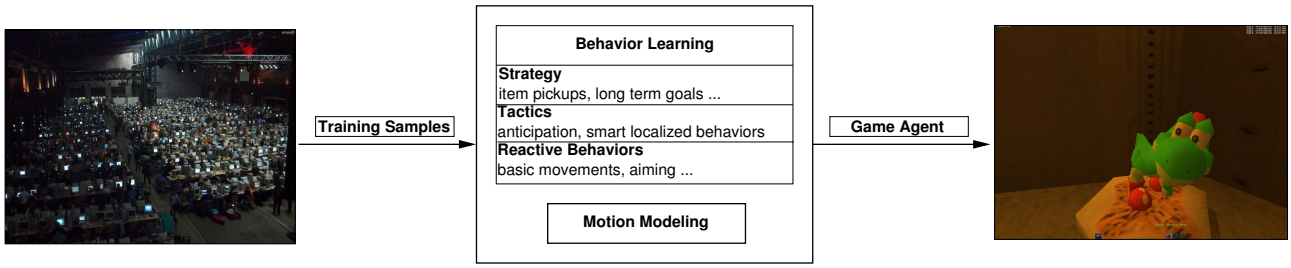

Figure 3: A model for generation of human-like game agents by imitating human play styles.

visualized in Figure 3 (since the model is rather abstract, it's transfer to other games is straightforward). The decision for this design was based on observations as well as on a widely believed psychological hierachy of human behaviour (Hollnagel 1994).

On top-most level, there are *strategic* behaviours which are used to achieve long term goals. In our case, besides the obvious goal of winning the game, several other (sub)goals were identified. Most strategies are targeted at securing important areas and important items. Still there is a constant change in specicific subgoals which depends on the current game-state (e.g. if a player is low on health, health refilling items might be more attractive than the most powerful weapon). The player who manages to control the important places of the map, with the most valuable items, will greatly enhance his chances to win the game.

The second layer represents *tactical* behaviours. Tactics usually are defined by a smart, localized situation handling. While the strategy tells the player about the next important region on a map, the tactics are responsible for evading possible threats on the way. Since tactics highly depend on anticipation of an enemy player's movement, a broader understanding of a scene is necessary. Prominent examples include the laying of traps or securing of areas by putting it under constant fire.

The most basic layer is given by *reactive* behaviours. Here we find simple reactions to audio-visual percepts. This includes movement, jumps, but also aiming and shooting on an enemy player as well as the prediction shots based on audible cues.

Although this discrimination of behaviours is rather strict and suggests a selection of one active layer of behaviours, they are to be understood as concurrent sets of behaviour. However, unexperienced players are more likely to concentrate on only one aspect. They will either engage in combat or look for better items, whereas experienced players have no problems in improving their strategic position while being in combat (often combat itself holds many tactical parts as well, e.g. taking cover or avoiding small passages). A classification of behaviours into categories makes the whole problem

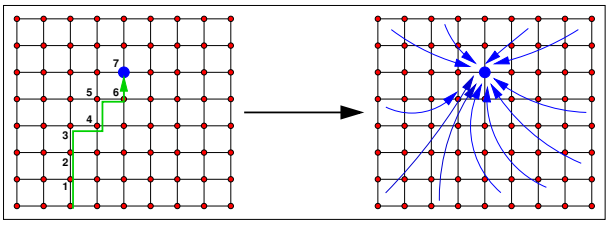

Figure 5: The left figure shows an observed movement pattern of a human player in the topological map representation, the right figure shows the corresponding computed potential field forces.

of imitating humans a lot easier and finally manageable.

Apart from behaviour learning we discovered that the problem of motion modeling is another integral part for imitation learning. In early experiments, we noticed that, although the actions seemed reasonable, the motion of our artificial agent looked jerky and were more robot- than life-like. Moreover there is a strong coupling between the actions a human player performs and the actions he is motion-wise capable of doing. Artifical agent's usually do not have such restrictions, they can instantly turn and even aim perfect on long ranges. Therefore the imitation of a human controlled agent's motion is necessary and poses another challenge.

On the one hand, we thus need to find approaches for learning by observation what to do in a given game situation. On the other hand, we have to find the appropiate, life-like motion for doing so. An integration into a consistent framework would be desirable.

Learning Strategies

In (Thurau et al. 2004*a*), we presented an approach for learning strategic behaviours. It realized goal oriented movement in 3D, including situation dependent item pickups and area control.

The approach consists of two parts. First, a topological representation of the 3D gaming world is learned by means of applying a *Neural Gas Algorithm* to the positions $\vec{p} = [x, y, z]$ a human player held during a match. Neural Gasses extended the popular k-means clustering algorithm and are especially suited for building topological representations (Martinez & Schulten 1991). In our case, the result is similar to the widely used waypoint-maps. But since it is data-driven, it provides an elegant and accurate way of generating waypoint maps corresponding to human gameplay. Figure 4 shows an exemplary topological representation.

In a second step, *Artificial Potential Fields* are placed into the topological map. These potential fields then guide the game-bot. Since strategies change according to game-states, the training samples are clustered in the state space of the agent yielding prototypical game states. Here the game-state includes information about the current items of the player and his health and armor values.

For each such state a potential field force distribution is computed which recreates movement patterns typically observed in that state, Figure 5 shows a simple example. Changes in the internal state of the agent cause switches among the field forces and thus will lead the agent to other, more attractive locations and items, thereby implicitly defining situative sequential item pickups as long-term goals.

To cope with known flaws of potential field approaches, namely local minima and weak potential field forces on certain parts of the map, we make use of *Avoiding the Past Pheromone Trails* (cf. (Balch & Arkin 1993)). These trails reinforce a chosen direction and drive the agent out of local minima. The obtained results were convincing. Strategic behaviours, situation dependent item-pickups or preferences for certain map areas could be learned and convincingly reproduced.

Learning Tactics

Tactical behaviours were described as a smart localized behaviour. However, learning or imitating such behaviours is not an easy task. It requires a broader understanding of a situation. For example, the observation of a human player ambushing an enemy player is itself a rather simple sequence of actions. But in order to imitate it, it wouldn't be enough to copy the action sequence. Instead, underlying contextual prerequisites to activate tactical behaviour have to be extracted. In the given example the enemy player might have been entering a room with only one exit, thereby giving the opportunity to ambush him. But learning such an understanding for situations provides a great challenge.

Right now we are searching for an appropriate representation of such more abstract scene understanding. Unfortunately, up to now there are only little known techniques of how to approach that topic. Moreover, since we prefer a data-driven approach to the imitation of tactical behaviours, we do not want to label sample sequences or rely on common game AI methods like finite-state machines or scripting. Recent work thus examined the application of *Mixture of Experts* architecture (Jordan & Jacobs 1994). In particular, we studied the context dependent handling of different weapon types. First results are encouraging, i.e. Mixture of Experts architectures were observed to learn the handling of different weapons; further experiments are necessary though.

Learning Reactive Behaviors

In (Thurau et al. 2003), we presented an approach for learning purely reactive behaviours (note that potential fields are often referred to as *reactive* behaviours, but since we effectively model long-term goals with them, we discussed them above). In QUAKE II®, reactive behaviours can be characterized as a direct functional mapping of game states $\vec{s}_t, \vec{s}_{t-1} \ldots \vec{s}_{t-n}$ onto player reactions $\vec{a}$. Typical reactive behaviours include aiming, shooting or dodging projectiles.

At first the training sample set is separated by letting a *Self Organizing Map* (SOM) (Ritter et al. 1992) unfold itself into

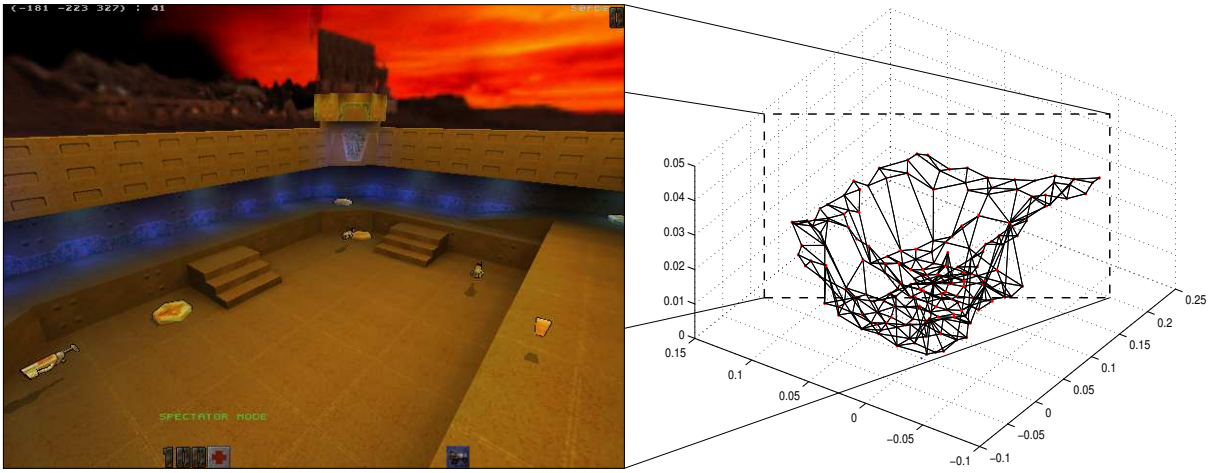

Figure 4: A 3D map and its topological representation as an outcome of a Neural Gas algorithm

the game-state space (in (Thurau et al. 2003), the game-state space consisted of the agent's position and an enemy's position). Training samples are then assigned to a corresponding SOM Neuron thus separating the data into different clusters. In a second step, we trained *Multi Layer Perceptrons* (MLP) for each cluster. Figure 6 provides a graphical presentation of the described approach.

For each new game situation, the most similar prototypical game-state from the SOM-Neurons is selected and its corresponding MLP used for behaviour generation.Thereby simple reactive behaviours, aiming or movement in 3D, could be learned. However, even after introducing time-dependent *Time-Delayed-Neural-Networks* the approach is limited to ad-hoc reactions to game-states. Long-term goals and planning of actions should be taken care of by strategical or tactical behaviour models.

Motion Modeling

Although the approaches presented so far reproduce human behaviours, the game agent's motion often appears jerky and can be easily distinguished from the smooth motions of a human player. An artifical game agents motion is thus an integral part of its appearance, for a realistic impression it should be as human-like as possible.

Our approach to motion modeling from imitation is based on movement-primitives (Fod et al. 2002). Movement-primitives, as the basic building blocks of movement, can be derived from a game agent's motion vectors using *Principal component analysis* (PCA). Thereby individual movement-primitives for individual players are obtained. To discretize a player's motion, prototypical motion vectors are extracted from the projections of the training-sample motion vectors onto the eigenvectors (or movement-primitives), using a k-means algorithm. Every single training sample motion vector can now be described by a more general action primitive $\vec{v}$. Usually a number of up to 800 action primitives is sufficient and covers the set of possible motions very well. Choosing less action primitives results in a more choppy movement, however, larger numbers of action primitives didn't lead to an observable smoother recreation of movements.

Complex movements result from sequencing action primitives. Since the *right* sequencing of action primitives, for imitating a human controlled game agent's motion, is given in the training sample set, probabilities for the execution of an action primitive can be extracted. Two transition matrices are computed. One expresses interdependencies between action primitives, the other expresses localized dependencies, based on the position of the player in the topological representation.

Given these matrices, the next action primitive to execute is chosen according to a roulette wheel selection over the probabilities for all action primitives. The probability for the execution of a single primitive $\vec{v}_i$ can be denoted as:

$$P_{\vec{v}_i} = \frac{P(\vec{v}_i|\vec{v}_l, w_k)}{\sum_{u=1}^{n} P(\vec{v}_u|\vec{v}_l, w_k)} = \frac{P(\vec{v}_i|\vec{v}_l)P(\vec{v}_i|w_k)}{\sum_{u=1}^{n} P(\vec{v}_u|\vec{v}_l)P(\vec{v}_u|w_k)}$$

where $P(\vec{v}_i|w_k)$ denotes the probability of executing action primitive $\vec{v}_i$ for topological node graph node w_k, and $P(\vec{v}_i|\vec{v}_l)$ denotes the probability of executing action primitive $\vec{v}_i$ as a successor of action primitive $\vec{v}_l$. These probabilities can be extracted from the training samples by inspecting the observed action primitive sequence. Since all probabilities can be computed in advance, the approach is computationally rather inexpensive, managing to execute up to 20-30 action primtives a second in our MATLAB® QUAKE II® client. A detailed analysis can be found in (Thurau et al. 2004*b*).

Using this approach, we managed to imitate (or recreate) complex sequences of motion. This includes not only simple movements, but also jumps over ledges and the infamous rocket jump (a maneuver, where a player fires a rocket on the ground and jumps at the same time, thereby reaching otherwise unreachable places – considered an experienced

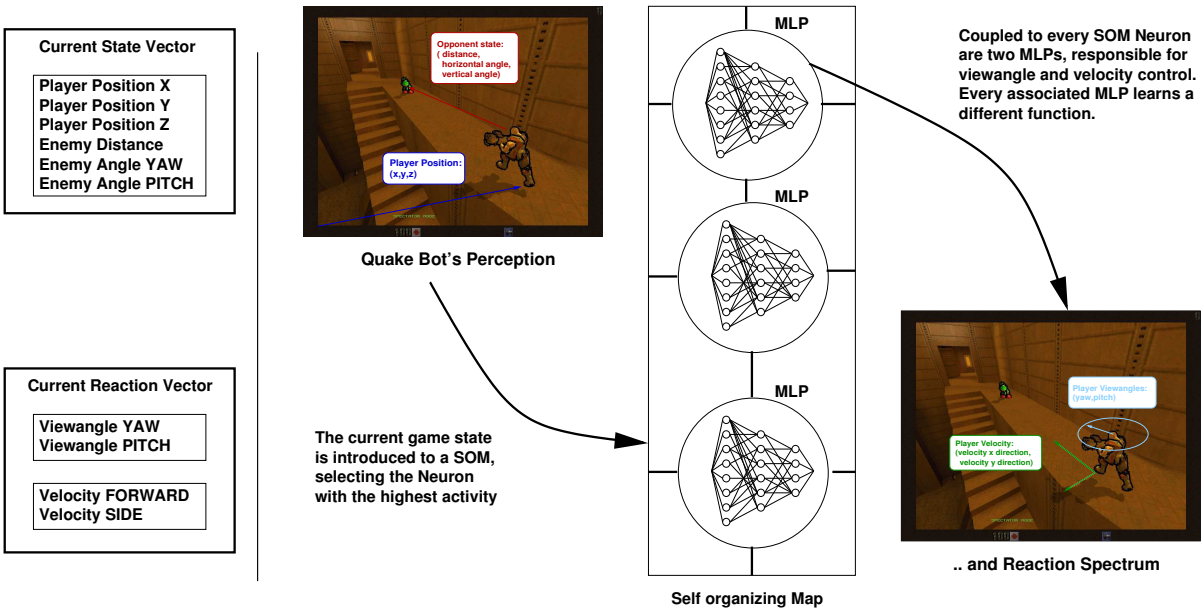

Figure 6: Architecture for learning situation dependent reactive behaviour, establishing a direct mapping of state space variables onto player actions.

Figure 7: Game-agent performing a long jump by means of movement primitives

player's move). Figure 7 shows the artificial agent performing a long jump to an otherwise not reachable region. The created movements appeared smooth and payed attention to the observed human player's style of movement, creating indeed life-like motion.

Conclusion

In order to create more human-like computer game agents, we proposed the usage of imitation learning. Thereby following a general trend in robotics towards imitation learning and following evidences in psychology on the importance of imitation for behaviour development in infants. Unlike other approaches in the field of machine learning in games, we concentrate on the highly complex genre of Ego or First-Person-Shooter games. These game types provide us with an incredibly huge amount of training samples - records of human player's, being downloadable from the Internet. We presented a comprehensive approach for the imitation of a human controlled game agent for a FPS game.

First, we identified different behavioural layers, namely strategic, tactical and reactive behaviours. Besides the behaviour learning, we clarified the importance and influence of motion modeling on a life-like appearance of an artificial game character. Finally, we outlined suited approaches for behavioural learning in each of the mentioned layers and sketched our recent work on the topic of motion modeling.

Since human behaviour during a game is very complex and rich of problems when it comes to machine learning, an artificial player that integrates all the techniques presented above was not yet realized. However, from the results discussed in this paper, it is reasonable to conclude that imitation learning is a well suited method for behaviour generation of artificial game characters. Concerning individual aspects of game

play, our game-bots outperform conventional bots which are driven by finite state machines or similar architectures. Bots that learned by imitation definitely stay closer to what a human player is doing since they consequently rely on observations of human players. Therefore, we are convinced it is worthwhile to further pursue this topic in order to see how far imitation learning will bring game characters.

Acknowledgements

This work was supported by the German Research Foundation (DFG) within the graduate program "Strategies & Optimization of Behaviour".

REFERENCES

Amir, E. & P. Doyle. 2002. Adventure Games: A Challenge for Cognitive Robotics. In *Proc. Int. Cognitive Robotics Workshop*. Edmonton, Canada: .

Balch, T. & R. Arkin. 1993. Avoiding the past: A simple but effective strategy for reactive navigat. In *Proc. IEEE Int. Conf. on Robotics and Automation*.

Bauckhage, C., C. Thurau & G. Sagerer. 2003. Learning Human-like Opponent Behavior for Interactive Computer Games. In *Pattern Recognition*. Vol. 2781 of *LNCS* Springer-Verlag.

Fod, A., M.J. Mataric & O.C. Jenkins. 2002. "Automated Derivation of Primitives for Movement Classification." *Autonomous Robots* 12(1):39–54.

Hart, P.E., N.J. Nilsson & B. Raphael. 1968. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths." *EEE Trans. on Systems Science and Cybernetics* 4(2):100–107.

Hietanen, J.K. & D.L. Perrett. 1996. "Motion sensitive cells in the macaque superior temporal polysensory area: response discrimination between self-generated and externally generated pattern motion." *Behavioral Brain Research* 76:155–167.

Hollnagel, E. 1994. *Human Reliability Analysis: Context & Control*. Academic Press.

Jordan, M. I. & R. A. Jacobs. 1994. "Hierarchical mixtures of experts and the EM algorithm." *Neural Computation* 6:181–214.

Kohler, E., C. Keysers, M.A. Umiltà, V. Gallese L. Fogassi & G. Rizzolatti. 2002. "Hearing Sounds, Understanding Actions: Action Representation in Mirror Neurons." *Science* 297:846–848.

Laird, J.E. 2001. "Using a Computer Game to develop advanced AI." *IEEE Computer* pp. 70–75.

Le Hy, R., A. Arrigioni, P. Bessière & O. Lebeltel. 2004. "Teaching Bayesian behaviours to video game characters." *Robotics and Autonomous Systems* 47(2–3):177–185.

Martinez, T. & K. Schulten. 1991. A Neural Gas Network learns Topologies. In *Artificial Neural Networks*. Elseviers Science Publishers B.V.

Mealy, G.H. 1955. "A Method for Synthesizing Sequential Circuits." *Bell System Technology J.* 34:1045–1079.

Moore, E.F. 1956. Gedanken-experiments on Sequential Machines. In *Automata Studies*. Number 34 *in* "Annals of Mathematical Studies" Princeton University Press pp. 129–153.

Nareyek, A. 2004. "Computer Games – Boon or Bane for AI Research." *Künstliche Intelligenz* pp. 43–44.

Rao, R.P.N. & A.N. Meltzoff. 2003. Imitation Learning in Infoants and Robots: Towards Probabilistic Computational Models. In *Proc. AISB 2003 Convention: Cognition in Machines and Animals*. Aberystwyth, UK: .

Ritter, H., T. Martinetz & K. Schulten. 1992. *Neural Computation and Self-Organizing Maps*. Addison-Wesley.

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

Schaal, S., J. Peters, J. Nakanishi & A. Ijspeert. 2003. Learning Movement Primitives. In *Proc. Int. Symposium on Robotics Research*. Siena, Italy: .

Sklar, E., A.D. Blair, P. Funes & J. Pollack. 1999. Training Intelligent Agents Using Human Internet Data. In *Proc. 1st Asia-Pacific Conference on Intelligent Agent Technology (IAT-99)*.

Spronck, P., I. Sprinkhuizen-Kuyper & E. Postma. 2002. Improving Opponent Intelligence through Machine Learning. In *Proc. 3rd Int. Conf. on Intelligent Games and Simulation (GAME-ON'02)*.

Thoroughman, K.A. & R. Shadmehr. 2000. "Learning of action through adaptive combination of motor primitives." *Nature* 407:742–747.

Thurau, C., C. Bauckhage & G. Sagerer. 2003. Combining Self Organizing Maps and Multilayer Perceptrons to Learn Bot-Behavior for a Commercial Computer Game. In *Proc. GAME-ON*.

Thurau, C., C. Bauckhage & G. Sagerer. 2004*a*. Learning Human-Like Movement Behavior for Computer Games. In *Proc. 8th Int. Conf. on the Simulation of Adaptive Behavior (SAB'04)*.

Thurau, C., C. Bauckhage & G. Sagerer. 2004*b*. Synthesizing Movements for Computer Game Characters. In *Pattern Recognition*, ed. C.E. Rasmussen, H.H. Bülthoff & et. al. Vol. 3175 of *LNCS* Springer.

Yannakakis, G. N. & J. Hallam. 2004. Evolving Opponents for Interesting Interactive Computer Games. In *Proc. 8th Int. Conf. on the Simulation of Adaptive Behavior (SAB'04)*. pp. 499–508.