Teaching Agents to Drive
CPSC 533B Topics in Computer Graphics: Algorithmic Animation

Asher Lipson, #25838020
18 April 2003

Abstract

This report presents the results of a course research project for Algorithmic Animation. A probabilistic model approach is taken to solving the problem of an agent driving on a race track and dodging obstacles. The model uses game logs from humans driving on the race track in order to try and get the agent to emulate human-like play. Results are presented for the agent driving the same track that humans drove on as well as for the agent driving a track different to that which the humans drove on.

1 Introduction

This report presents the results of a research project for the CPSC 533B course, Algorithmic Animation. The problem being looked at is race track driving. In particular, one wants to create a software agent that is able to drive around different race tracks. Each track contains a number of obstacles which the agent should be able to manoeuvre around.

Giving an agent the ability to drive around a race track in a human-like manner would be extremely useful to the computer gaming industry, where creating realistic artificial intelligence (AI) is seen as being a key feature to the industries growth [Champandard, 2002]. This research uses computer games as a platform for AI learning, something that has been previously proposed [Laird and Duchi, 2000].

This problem can be considered an extension to that looked at in [Lipson *et al.*, 2003], where an agent learnt to play a room exploration game based on previous human plays of the same game. This research takes things further by applying the human game data from one track and allowing an agent to race on a different track. From the human generated data, a probabilistic mixture model is learnt using the Expectation Maximisation (EM) algorithm. This model allows for querying; choosing an action for the agent based on what humans did in similar circumstances.

This research differs from a number of similar works in that it does not seek to create an optimal agent that plays a track perfectly. This problem has been looked at previously and there are a number of existing solutions that do relatively well in creating optimal game play. Obtaining human like behaviour (which often is not optimal), includes allowing the agent to make mistakes and to make moves that do not make sense, much as humans do. In addition, the agent should not have more knowledge of the world than a human would have under similar circumstances. The agent should not be globally aware nor omniscient and this leads to the use of a "lookahead window", where the agent is aware of its environment for some particular distance ahead of the vehicle.

To complement this report, a number of movies showing the human and agent playing the track game are available for download off the project website:
`http://www.cs.ubc.ca/~alipson/projects.html`.

The remainder of this report is structured as follows; the next section provides some background to the problem and briefly mentions related work that has been done. Section 3 describes the testing platform and the state representation that is used. Thereafter, the approach used to try and solve the problem is described in depth and this includes a description of the probabilistic model. In section 5, all the results for both human and agent game play are presented and these are discussed in section 6. Lastly, problems that were found and possible future work is mentioned in section 7 and this report is concluded with section 8

2 Background

The first problem that comes up in any agent learning environment is how the agent should interact with the environment around it and how to represent its knowledge of the world. There has been some work done on automatically learning what information is important, based on the readings from a number of sensors. There is however the implicit assumption that some of the sensors duplicate information and can be ignored in favour of other sensors. It also means that the agent has access to information from all sensors and that this contains all data. This does not solve the problem of what information the sensors provide. In particular this has been done in a vehicle environment [Koike and Doya, 1999], where the sensors provided information on various aspects of the vehicle.

There is a large field of research into software agents and their potential in a number of different applications. A number of survey papers and books [Nwana, 1995], [Russel and Norvig, 1995] have been written on this field and for a thorough review of this area, the reader should consult one of these.

Much of the other related work revolves around using some form of reinforcement learning with rewards in order to drive a path on a track. This is often combined with the aim of learning an optimal path according to some conditions such as shortest time or least expended energy. A common optimal policy in use is for the agent to try and reach as high a speed as possible and to then either brake in the face of slower traffic or to overtake them.

[Barto *et al.*, 1993] learn a path that is optimal (according to their

criteria) on a short track with no obstacles. Work has also been done in using Icarus, an agent architecture with reinforcement learning embedded, to learn a highway driving model [Shapiro *et al.*, 2001]. Research has also been done in trying to model a human's learning ability in car driving with a neural network [Wewerinke, 1994]. This includes overtaking and lane keeping through a system theory approach.

The research described in this report differs in that it aims to create an agent with driving abilities similar to a human. The creation of an agent that can drive optimally is not a goal.

[Lipson *et al.*, 2003] described an approach by which the agent based its actions and decisions purely on the data generated by humans in the same simulation that the agent was playing. This allowed the agent to effectively "mimic" the actions of a human. A probabilistic model similar to the one used in that research is applied here. The biggest difference between [Lipson *et al.*, 2003] and the research described here is in the complexity of the environment and the testing of the agent in worlds different to that used for the human game play.

3 Representation

3.1 The testing platform: Track game

The testing platform for this research is a racetrack which the player (human user or game agent) drives a vehicle on. There is a single vehicle with the human user having a top down view of the world and the track scrolling vertically down from the top of the screen to the bottom. A human's view of the game showing a track with no obstacles can be seen in figure 1 and a track with obstacles in figure 2.

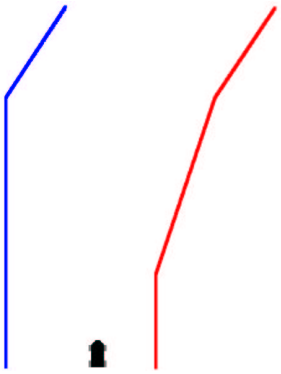

Figure 1: The human's view of the game

Both the human user and the agent are aware of the track state and any obstacles in a window ahead of their current distance. This window, referred to as the lookahead can be seen in figures 1 and 2, where the human user can see the track area for a distance of 5 points ahead of its current position. The testing in section 5 was done with a lookahead of 5 and 10. Each track used had 50 points (rows) to it and looped around so that the end of the track joined back up with the start of the track - this allowed for the player to race a number of laps.

The world is considered to be non-deterministic, with the same action in the same state at different times allowed to lead to different states [Kaelbling *et al.*, 1996]. The vehicle has a minimum speed of zero (stationary) and a top speed of three. Trying to increase the current speed above three or below zero keeps the vehicle's speed at that boundary value. The width span of possible movement for the

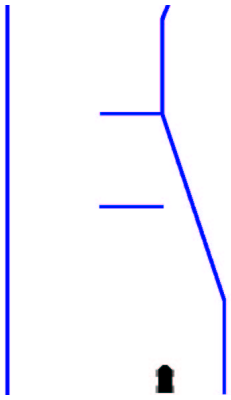

Figure 2: Obstacles ahead of the vehicle

vehicle is sixteen, thus from the vehicle's starting position, one can move a maximum of eight positions to the left or right. This prevents the vehicle from going off screen. As with the speed, trying to move beyond these bounds had no effect.

To emphasise the racing track paradigm, the vehicle is penalised for leaving the track. The first time the vehicle leaves the track, its speed is reduced to zero. One can imagine this as being a car stalling when it leaves the road. To get the vehicle moving again, the player has to increase speed. If the player continues being offtrack (without returning to the track), no speed reduction is done. However, if the player returns to the track and then goes offtrack again, the speed is changed to zero. Thus each time the player leaves the track, they are penalised. One would thus want to stay on track as much as possible.

Obstacles are implemented as horizontal "bars" that the vehicle cannot cross through, but has to move around. In addition, the area in front of an obstacle is slippery, meaning that the higher the vehicle's speed, the harder it is to avoid careening into the obstacle. If the player hits an obstacle, the vehicle's speed is cancelled out to zero and no further increase in speed can be made until the vehicle moves to the side of the obstacle. Each obstacle is two horizontal points wide, meaning that the agent can hit the obstacle in three places; the center of the obstacle or the left and right edges.

The player has a choice of five actions at each time step; increasing or reducing the speed, shifting left or right and doing nothing. Doing nothing essentially means no formal action of changing position or speed is made and this is considered to be a valid move.

3.2 State representation

In order to model a human's view of the world, the agent's state at each time step is defined as containing three pieces of information, the current speed, whether there is a crash ahead and the distance to the crash. The crash ahead can take three values, -1 indicating the crash will be with the left side of the track, 0 implying no crash ahead and 1 meaning that the crash will be with the right edge of the track. The vehicle can only detect a crash from inside the track, so moving from a position off track to a position on track across a track edge does not register a crash. A similar representation is provided for the obstacles. The state contains information on the closest obstacle directly in front of the vehicle (i.e. obstacles to the side are not recorded in the state). This information contains two information bits, whether there is an obstacle in the immediate line of sight straight ahead and how far that obstacle is from the vehicle. Thus there are two potential crash possibilities for the agent, the track edge, for which the vehicle is not penalised and an obstacle, which the vehicle cannot drive through, but has to move around.

The distance measure for a track edge crash has a range of [-1, (window size - 1)]. -1 implying that there is no upcoming crash, 0 meaning that there is a crash at the present time step and a positive value indicating the distance to the crash. Thus a crash can be recorded as taking place in "window size" possible positions ahead of the agent (including at 0). The distance is measured relative to the agent, so one could view the agent as being at position (0, 0) in its local coordinates. The agent is only aware of the closest crash to its current position. Thus if the vehicle is going to crash in one time step with the right track and five time steps with the left track, then the current state would be [speed, 1, 1], indicating the nearest crash is one time step away with the right track edge. With the window size incorporated into the distance measure, one ensures that the agent is not aware of possible crashes ahead of the window, much like a human cannot tell what the track looked like outside of the window. The obstacle measurement contained whether there is an obstacle ahead and the distance to that obstacle. A distance of 1 indicates the vehicle has crashed into the obstacle.

The vehicle's speed is in the range [0, 3], there are three possible crash descriptions for the track edge [-1, 1], six possible distance to crash measurements (for the track edge) [-1, 4] (for a lookahead of 5). For the obstacles, there are two values $\{0, 1\}$ for the crash indicating an obstacle ahead or not and six possible distances, again [-1, 4] taking into account the lookahead window size.

Originally, the state was going to contain information on all track edges and obstacles in the lookahead window, but this meant that the state space was large and would quickly become unwieldy. Hence the much simplified state representation containing information only on the nearest track edge and obstacle.

4 Approach

The approach to solving this problem starts with human generated data. Humans play a number of games and the data from those games is logged into files. Once games have been played, one is able to generate a probability matrix $G(a,\ s,\ k)$ of the action a made in state s for game k based on the game data. That is for each game we have the probability of actions being taken for specific states. The G matrix only contains information on states that are seen by humans. If no humans see a specific state during their game play, that state does not appear in any of our probability matrices.

This matrix is used as part of the Expectation Maximisation (EM) algorithm. The EM algorithm is used to cluster the various games into groups. EM is a fairly versatile algorithm allowing different data to be clustered together, including text, images and sound [Brochu *et al.*, 2003]. By clustering the games into groups we have an automatic mechanism for finding similar games. This potentially could be done by hand, but it would take an extremely long time. The EM algorithm is able to learn a model that describes all the data submitted provided to it. The model is outlined further in the next section, 4.1 and the EM algorithm in section 4.2. The two components that make up the model are $p(c)$, the probability of a cluster being used and $p(a|s,\ c)$, the probability of taking action a in state s, whilst in cluster c. These two measures provide an indication of how much of the game data each cluster contains as well as what the relationships are between states and actions within each cluster. EM does however converge to a local maximum and thus for each run of EM slightly different clusterings appear.

Once the EM algorithm is complete and clustering has been done, we choose a cluster through sampling from $p(c)$, i.e. we would normally end up in the cluster with the highest percentage of data, but will occasionally end up in clusters that hold less data (i.e. contain games that were less played by humans).

Thereafter, at each time step we take the agent's current state and match it to the closest state in our chosen cluster. The closest state is found by minimising the error measure:

$$\Delta_e \ = \ |s \ - \ \bar{s}|$$

where s is one of the states in the current cluster and $\bar{s}$ is the agent's current state.

We then sample an action from the generated $p(a|s,\ c)$. We again would obtain actions that are most often made by humans in that state, but will also choose actions that are made less often by humans. This allows us to obtain actions from the human generated actions distribution.

The next two subsections, 4.1 and 4.2 provide more detail on the model learnt by EM. Note that the next two sections do not go into the full details of the algorithm or the expressions, but rather try to provide enough information to gain an understanding for this portion of the report.

4.1 Model Specification

Each game is represented as a table $G_k(k \ = \ 1, \ldots, n_g)$, n_g representing the number of games played by humans. Each row represents a game state and each column a specific action.

The mixture model is described as:

$$G_k|\theta \ \overset{iid}{\sim} \ \sum_{c=1}^{n_c} p(c) \prod_{a=1}^{n_a} \prod_{s=1}^{n_s} p(a|s,c)^{G_{ask}}$$

with $\theta \triangleq \{p(c), p(a|s,c)\}$ representing the model parameters. n_c is the number of clusters (chosen by the user), n_a is the number of possible actions (for the track game there are five possible actions) and n_s is the total number of states (determined by the number of games played by humans). We have that $\sum_{c=1}^{n_c} p(c) \ = \ 1$. For this research, $n_c \ = \ 10$, which is the maximum number of possible clusters. One can still have clusters that contain no games, eg. games are split into only four clusters and thus six clusters contain no information.

4.2 MAP Estimation with EM

EM consists of two steps, an E step and an M step. The E step updates our expectation with respect to the model parameters $p(c), p(a|s,c)$ and the M step updates our model parameters. These two steps are continually run one after another, updating the expectation and model parameters. One can either choose to run the steps until there is little change in value between two subsequent steps, or can run for a set number of iterations. In this research, the steps were run for 40 iterations. When updating $p(c)$ in the M step, one is essentially calculating which cluster games should be placed in. The MAP estimation does not try and fit to all clusters, but allows some clusters to contain no games - i.e. only the essential clusters are used.

In our M step, the following two expressions are used to update $p(c)$ and $p(a|s,c)$:

$$\widehat{p}(c) \ = \ \frac{\alpha \ - \ 1 \ + \ \sum_{k=1}^{n_g} \xi_{ck}}{\sum_{c'=1}^{n_c} \alpha \ - \ n_c \ + \ n_g}$$

$$\widehat{p}(a|s,\ c) \ = \ \frac{\beta \ - \ 1 \ + \ \sum_{k=1}^{n_g} G_{ask} \xi_{ck}}{\sum_{a'=1}^{n_a} \beta \ - \ n_a \ + \ \sum_{a'=1}^{n_a} \sum_{k=1}^{n_g} G_{a'sk} \xi_{ck}}$$

with ξ_{ck} the expectation for cluster c and game k. β and α are Dirichlet prior parameters that force down the number of clusters, that is try and prevent to many clusters from being used. In the EM runs, α and β are given the values: $\alpha \ = \ 1.1$ and $\beta \ = \ 5$.

5 Results

Each of the results described in this section are for an agent or human doing two laps around the track. At the end of the two laps the vehicle also has to be on the track, i.e. if the player completes two laps, they would only register as complete when the vehicle is on the track.

When referring to the track that the agent is trained on, it refers to the track that the human player used. That is the track from which the human data is generated, in order to construct the probability matrices.

Table 1 presents overall information for the human player for two different lookaheads on the track shown in figure 3, the training track. Table 2 provides some more detailed statistics on the same human runs in terms of the minimum, maximum and average value for each of the measurements. Results for the agent racing the same track as the human are provided in table 3 and for a different track in table 4. This section presents the results without discussion. Section 6 compares and discusses the results between the various tables.

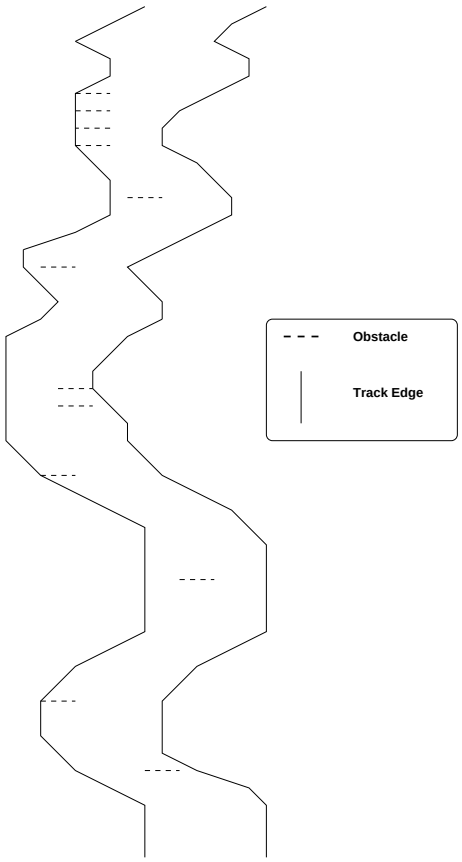

Figure 3: A full view of the training track played by both the humans and the agent

5.1 Human training data

In table 1, for each of the two lookaheads, ten games were played. `Total states` measures the number of states that were saved over the course of the ten games. One can see that with a lookahead of 10, a smaller number of states are saved. The number of states is an indication of how long the humans took to complete the laps, with less states indicating faster completion. `# times offtrack` measures the number of times that the vehicle went offtrack and `Time spent off` measures the amount of time that the vehicle stayed off track. A player could leave the track once, but stay off track for 4 time steps. Lastly, `Obstacles hit` measures the amount of time that the vehicle spent stuck behind an obstacle. A human could spend a number of time steps stuck behind an obstacle, before they moved around it and continued on the track.

Look ahead	Games	Total states	# times offtrack	Time spent off	Obstacles hit
5	10	1442	8	16	96
10	10	1271	7	10	72

Table 1: Human Training Data Total

		Lookahead	
		5	10
1	Total Time$_{avg}$	142.3	125.2
2	Total Time$_{min}$	131	105
3	Total Time$_{max}$	155	152
4	# times offtrack$_{avg}$	0.8	0.7
5	# times offtrack$_{min}$	0	0
6	# times offtrack$_{max}$	3	2
7	Time spent off$_{avg}$	1.6	1
8	Time spent off$_{min}$	0	0
9	Time spent off$_{max}$	5	2
10	Obstacles hit$_{avg}$	9.6	7.2
11	Obstacles hit$_{min}$	1	0
12	Obstacles hit$_{max}$	21	15

Table 2: Human Training Data Bounds and averages

For each main measurement in table 2, the minimum, maximum and average values are provided as an indication of the human bounds. Each statistic corresponds to the measurement of the same name in table 1. One can see that the larger lookahead of 10 allowed the human to finish the laps quicker and hitting on average slightly fewer obstacles along the way. Both sets of games were rarely offtrack.

5.2 Training, racing on the same track

An agent was given 1000 time steps in which to complete two laps. Table 3 shows the results for an agent racing the same track that the human data was obtained from. These results are for the agent playing ten games. The statistics are for a lookahead of 5 and 10. The track was that shown in figure 3. The rows correspond to the measurements described in section 5.1 with the minimum , maximum and average value shown for each measurement and each of the two different lookahead values.

5.3 Training, racing on different tracks

The statistics in table 4 are for an agent using the human data from the original track (figure 3) and racing on a new track. The new track, shown in figure 4 is the same length as the original track and also contains obstacles, but is different in that it is a lot thinner in width (on average half as wide). This was done to test the scalability of the method and to see whether one could apply data gained from

		Lookahead	
		5	10
1	Total Time$_{avg}$	303.9	288.4
2	Total Time$_{min}$	131	150
3	Total Time$_{max}$	476	379
4	# times offtrack$_{avg}$	14.2	13.6
5	# times offtrack$_{min}$	4	5
6	# times offtrack$_{max}$	28	19
7	Time spent off$_{avg}$	143.6	140.1
8	Time spent off$_{min}$	84	91
9	Time spent off$_{max}$	222	167
10	Obstacles hit$_{avg}$	41	34.8
11	Obstacles hit$_{min}$	0	0
12	Obstacles hit$_{max}$	99	74

Table 3: Summary of results for agent racing and trained on the same track

		Lookahead
		10
1	Total Time$_{avg}$	343.6
2	Total Time$_{min}$	254
3	Total Time$_{max}$	470
4	# times offtrack$_{avg}$	16.2
5	# times offtrack$_{min}$	5
6	# times offtrack$_{max}$	29
7	Time spent off$_{avg}$	190.4
8	Time spent off$_{min}$	123
9	Time spent off$_{max}$	266
10	Obstacles hit$_{avg}$	91.3
11	Obstacles hit$_{min}$	56
12	Obstacles hit$_{max}$	177

Table 4: Summary of results for agent racing and trained on different tracks

one track to another track. This is often done by humans where they do not have a separate driving method for each road they are on, but are able to use what they know about one track on a different track.

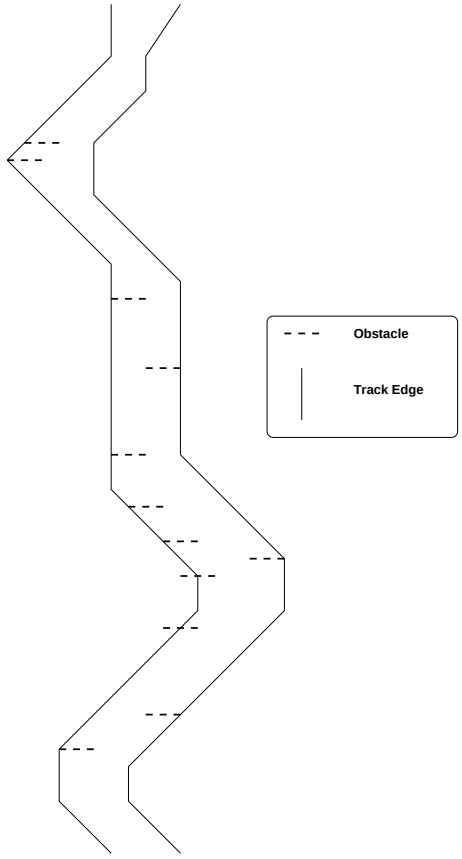

Figure 4: A full view of the test track played by the agent

The rows and descriptions presented in table 4 are the same measurements as those shown in the previous tables, with minimum, maximum and average values given. This table is a summary of the results for an agent playing ten games. Tests were only done for a lookahead of 10 due to that lookahead providing slightly better performance than a lookahead of 5 when racing on the same track it was trained on (table 3).

6 Discussion

There are a number of results that can be compared and discussed; agent vs. human on the same track, comparing different lookahead distances for an agent and finally looking at the performance of an agent on an unknown track. Section 6.1 will look at the agent performance compared to the human on the same track. The results for the same track, but different lookahead distances for the agent is discussed in section 6.2 and section 6.3 looks at how the agent fared on a new unseen track.

6.1 Human vs Agent on the same track

This section compares the results generated for the ten human played games presented in section 2 and for the agent played games for the two lookaheads in section 3. Both sets of results were for the same track.

The first thing one notices is the average time that it took to complete two laps of the track. The average time for both lookaheads is approximately half for the human player what it is for the agents. The agent took twice as long on average and even worse for the maximum time, where the agent was two and a half to three times slower in maximum time. Thus on average the agent did not complete the laps in a time even close to that of the human. However, for a lookahead of 5 the agent managed to achieve the same minimum time as the human once. The method thus can produce a good agent in terms of human measurements. Unfortunately this did not occur for a lookahead of 10, where the agent was approximately 50% slower than the human of the same lookahead.

The humans managed to stay ontrack extremely well and rarely ventured off track. This is not the case for the agents, where they frequently went offtrack. The humans left the track on average less than once per game, but the agents amounted to 14 times per game, a vast increase. The results decline further when one looks at the amount of time the agent spent offtrack. On average the agent spent over 100 times more time offtrack than the human. Each time the agent went offtrack it took about ten time steps to return to the track. This contrasts with the human approach of returning to the track within two or three time steps.

When humans played the game, what would sometimes happen is that they would increase their speed before returning to the track. They would also sometimes not make an action for several time steps before returning to the track. Thus when the agent had to choose an action it was from a distribution across mainly these three actions.

The number of obstacles hit appears to make the agent look extremely good at the minimum end of the scale and not so good at the top end. As in the offtrack measurements, the agent on average and at maximum spends more time hitting (or stuck behind) obstacles. Looking at the minimum bounds of the scale, it seems like the agent achieves as good, if not better results than the human player. However, there is some ambiguity in these results which can be explained through the offtrack numbers. Obstacles were only placed on the track and with the agent spending so much time offtrack it meant that there were no obstacles to hit. So the low obstacles hit value is due in large part to the agent not being on the track and thus there not being any obstacles to hit.

Although some of the total time values for the agent were competitive against human values, this was not the case for the obstacles or offtrack measurements.

6.2 The effect of the lookahead distance

Table 3 (page 5) shows results for a lookahead of 5 and a lookahead of 10. It was found in table 2 in section 5.1 that the larger lookahead allowed humans to complete the laps in a notability shorter time and hit slightly fewer obstacles - was the same evident in agents based on the human data?

A lookahead of 10 required on average a shorter amount of time to complete two laps and the maximum time was almost 100 time steps lower than that required by a lookahead of 5. However, strangely enough, the minimum time was shorter for a lookahead of 5 than 10. This was the same value that matched the minimum time for the human player. An explanation would perhaps be that EM managed to find an extremely good clustering for that run and that the agent was able to sample good actions out of those clusters.

We have a similar situation occurring in the number of times the agent went offtrack, a lookahead of 10 provided a slightly lower average number of times off track when compared to a lookahead of 5 and also a lower maximum value. However, the minimum value was in favour of a lookahead of 5, though only by a single time step.

This pattern again repeats itself for the time spent offtrack. The minimum time found across 10 games was lower for a lookahead of 5, than for a lookahead of 10, but a lookahead of 10 provided better results for the maximum and average amounts of time spent offtrack. Again, the maximum value was significantly lower for the lookahead of 10, compared to the lookahead of 5, with the minimum and average values being only slightly better for their respective lookaheads.

Looking at the obstacles hit measurements, the lookahead of 10 is better on average and at a maximum value than a lookahead of 5. The minimum number of obstacles for both lookaheads is 0, but as mentioned previously this is ambiguous as these cases were often for when the agent spent the majority of its time offtrack.

The biggest problem with both lookaheads is the amount of time they spent offtrack, compared to the total time they took to complete the track. The agent generally spent approximately half of it's time offtrack. That is a fairly woeful performance by any measure and needs to be corrected. When these figures are compared to the amount of time the humans spent offtrack out of the time they took to complete the laps, then it shows how bad the agent was actually doing.

The one positive in these results is that it showed that a lookahead of 10 definitely helped the agent achieve better results on average than a smaller lookahead. The minimum values for much of the measurement were better for the smaller lookahead, but this was for one specific case that worked extremely well. Overall, however, the agent had difficulty in staying ontrack for a meaningful amount of time.

6.3 Agent training and racing on different tracks

Table 4 (page 5) shows the results for the agent racing on a different track to that which the human data was obtained off. The humans drove on the track shown in figure 3 (page 4), whereas here, the agent drove on a different track, that shown in figure 4 (page 5). The track that the humans drove on will be referred to as the training track and the track that the agent drove on, the test track.

The agent took longer to complete two laps of the test track than for the training track. The major contribution to this was that both the minimum and maximum track times increased by 100 time points, 66% increase for the minimum and approximately 25% for the maximum. The maximum time for the lookahead of 10 on the test track was in fact less than the maximum time for a lookahead of 5 on the training track.

The minimum and average number of times offtrack shows little difference between the two tracks. However, the maximum number of times offtrack recorded by the agent was more than a 50% increase over the training track data. These values are only slightly (1 time step in the case of the minimum and maximum measurements) higher than those recorded by a lookahead of 5 for the training track.

The total time spent offtrack increases by a large amount over that recorded for the training track. Each of the minimum, maximum and average values increase by approximately 50%, with the maximum time spent offtrack (266) more than double that of the minimum time (123) for the test track.

The amount of time spent stuck behind obstacles is on average almost tripled from 34.8 for the training track to 91.3 for the test track. The test track no longer has cases where the agent managed to avoid all obstacles (as it did in the training track), but rather spent a minimum of 56 time steps stuck behind obstacles. The maximum value increased by 2.5 times over the agent's results for the training track. Part of this is expected. Although both the training and test tracks have the same number of obstacles, 12, the test track is much thinner and this makes it harder to avoid obstacles whilst still staying on the track.

The agent on the test track spends on average 55% of its total time offtrack. This is higher than the 49% that the agent spends offtrack on the training track. If one takes into account the amount of time on average that the agent spent stuck behind an obstacle, then 82% of the total time to complete two laps is spent either offtrack or ontrack, but stuck behind an obstacle. Again using the average values, the agent spends only 62 time steps on the track and not stuck. The agent definitely does worse on the test track with human data from a different track that it does on the training track with human data from that same track.

7 Problems and future work

A number of problems with the agent have been mentioned in previous sections. These are discussed in more detail with some possible solutions in this section.

7.1 Simplicity of the state

The information that is stored in the agent's state is fairly simplistic and tries to contain all the information that is needed for the track. This was perhaps part of the problem - too simple. One normally would want to abstract as much detail away from the state as one could and use the simplest possible state representation. However, in this case, the simplicity of the state meant that certain states that were visually different to the human eye, were the same state to the

agent. For example, as mentioned previously (section 3.1), the agent could hit an obstacle in three places (left edge, right edge and center of the obstacle), but the state does not differentiate between any of them. If a human is caught on the right edge of an obstacle then they would always move to the right, but to the agent there is no difference between being caught on the left, right or center parts of the obstacle – this would mean that actions for all three would be combined into actions for a single state even though the actions are for different states.

The simplistic state may also have partly contributed to the off-track problems. A human is able to discern that the state curves from right to left and that there are a number of possible crash positions ahead. The state for the agent only stored the closest possible crash point. Expanding this to include more than one possible crash point or to include all possible crash points within the lookahead window may be beneficial to this problem. However, this increases the state space dramatically. Excluding any obstacle information and only taking into account the first possible crash with the edges of the track for a lookahead of 5, there are 72 unique states. If one takes all track edge points into account within a lookahead window of 5, the number of states increases to almost 30 000. Including extra obstacle information takes this number to almost 360 000 states. A decision was made to keep the state as simple as possible and to reduce the number of possible states. However, based on these results, this may not have been the best choice. It is worth considering a more complex state containing more information.

More work is definitely required to look at a state containing different information and perhaps being representative of the entire environment within the lookahead window.

7.2 Unseen states and actions

The second problem again involves states. The model and probability matrices are constructed with states that the human has seen. If the agent reaches a state that was not seen by a human player, then the agent finds the closest state in the probability matrix and chooses actions from there. This would be especially important when one takes the human data from one track and uses it to control the agent on a different track.

The matching function was very simple and had each element of the state (speed, distance to crash, etc) weighted equally. Some work needs to be done in deciding whether certain elements are more important than others – this would affect which state was chosen for sampling when no exact state existed. An alternative solution would be to come up with a method for including states that the humans had not seen. Whether it is possible to infer a distribution over the actions for states that have not been seen is uncertain. One could possibly take the states that had been seen and find some form of interpolation or mapping method to take to states that had not been seen. This would change the problem of finding a good function to match to the closest state to a problem of inferring actions for a new state from previous states. The latter problem is arguably more difficult, but perhaps could be more rewarding to the results.

It may be possible to construct an action distribution for an unseen state based on previous states and action distributions. It may also have been previously worked on and some research needs to be done into this problem.

7.3 Agent history

The method as implemented only takes the current state into account with the agent not storing its history. It would be interesting to see if implementing some sort of history, perhaps in the form of a Markov process would affect the results. This could store a history of only one state or possibly more. One could alter the model to take this into account by using $p(s' \mid s,\ a,\ c)$ which would be the probability of ending up in state s' from state s when making action a and in cluster c. One could also change the games to include details on the previous state.

7.4 Other problems

As mentioned in previous sections, the agent spent almost half its time offtrack, which is far too much. It is possible that the problems outlined above could have contributed to this problem, but this is uncertain. Only by trying to solve them would this be found.

One idea that was proposed by Nando de Freitas was to combine this method with some form of reinforcement learning and perhaps to be able to switch between the two methods or to combine them. This is a very good idea which should be looked at in the future. However, it is felt that the method presented in this report should first be at an appropriate standard whereby the agents are competitive against humans before combining it with other methods.

8 Conclusion

This report presented the results of a research project into agent driving simulations. The aim was to construct a probabilistic model that would allow an agent to drive on a track with obstacles. The basis for the model would be game data that was generated by humans playing the same simulation.

A number of tests were done including the agent driving the same track that the human drove and the agent driving a different track to the humans. Results from these tests were discussed and comparisons made between the tests. Some discussion was also given on the effect of different lookahead windows, that is how much of the track and obstacles ahead of its current position the agent was aware of.

It was found that both a human and an agent benefited from having a larger lookahead, i.e. could see more of the track ahead. However, the agent was found on average to not be competitive against the human and produced results of a lower standard. There were some promising results though, including the agent producing a minimum time equal to that of a human for a lookahead of 5 on the same track.

Some possible problems that could have led to the low quality of the agent results were also presented, with possible solutions that could be implemented in the future. One of the hardest parts in creating an agent–environment interaction is deciding what information should be represented in the agent's state. One would normally look for the simplest possible state, but in this project it may have handicapped the agent.

Although much of the presented results were not what was hoped for, they led to the identification of a number of possible ways forward. These should lead to improvements in both this research project and similar work on agent track racing and human-like game play.

Acknowledgements

I'd like to thank Michiel van de Panne, David Poole and Nando de Freitas for their help and discussions on this project.

References

[Barto *et al.*, 1993] A.G. Barto, S.J. Bradtke, and S.P. Singh. Learning to act using real-time dynamic programming. Technical Report UM-CS-1993-002, University of Massachusetts, Amherst, 1993.

[Brochu *et al.*, 2003] E. Brochu, N. de Freitas, and K. Bao. The sound of an album cover: Probabilistic multimedia and information retrieval. In *AI-STATS*, Florida, USA, January 2003.

[Champandard, 2002] A.J. Champandard. The future of game AI: Intelligence agents. http://www.ai-depot.com, 2002.

[Kaelbling *et al.*, 1996] L.P. Kaelbling, M.L. Littman, and A.W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237 – 285, 1996.

[Koike and Doya, 1999] Y. Koike and K. Doya. Multiple state estimation reinforcement learning for driving model: Driver model of automobile. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, volume 5, pages 504 – 509, 1999.

[Laird and Duchi, 2000] J.E. Laird and J.C. Duchi. Creating human-like synthetic characters with multiple skill levels: A case study using the soar quakebot. In *AAAI 2000 Fall Symposium*, November 2000.

[Lipson *et al.*, 2003] A. Lipson, N. de Freitas, and E. Brochu. The touring test: Human-like play in computer games. Unpublished research paper, January 2003.

[Nwana, 1995] H.S. Nwana. Software agents: An overview. *Knowledge Engineering Review*, 11(2):205–244, 1995.

[Russel and Norvig, 1995] S.J. Russel and P. Norvig. *Artificial Intelligence, A Modern Approach*. Prenctice Hall, Upper Saddle River, NJ, 1995.

[Shapiro *et al.*, 2001] D. Shapiro, P. Langley, and R. Shachter. Using background knowledge to speed reinforcement learning in physical agents. In *The Fifth International Conference on Autonomous Agents*, Montreal, Canada, May 28 - June 1 2001.

[Wewerinke, 1994] P.H. Wewerinke. Modeling human learning involved in car driving. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, volume 2, pages 1968 – 1973, 2 - 5 October 1994.