# Postmortem for Q-BMW

By: Emuye Taylor and Brandon Lees

## Introduction:

### *Vision:*

Current first person shooters include some of the most popular games to date amongst general gamers. Some important titles are, Quake, Halo, Half-Life, and Doom. Although all of these games use different approaches to create the AI they feature, this AI has one thing in common. Current first person shooters feature very static AI. Many traditional approaches to AI for these games are based on a Finite State Machine (FSM) design. This approach models the AI's functionality (Wander, Attack, Retreat, Chase, etc) as states in an FSM. In order to create more complicated AI, more states are added to the FSM to represent more fine grained actions. From a player's point of view, the shortcoming of this type of approach is that regardless of how the player chooses to play the game or the difficulty level of the game, the AI will play the same. Another method used to make more challenging AI is to allow the AI to perform actions which the player is unable to. For example, the AI might be given a speed bonus or a damage bonus. Or, the AI might be able to shoot with an accuracy which is at a higher level then a gamer could achieve. These methods present an additional shortcoming to the static AI of current first person shooters. At higher levels of difficulty, for the player, the game play can feel as if the AI is cheating.

### *Goals:*

To address the problems with current first person shooter AI, we planned to implement a more dynamic AI which would be based on learning. The goal of this implementation was to offer a game which provided a more realistic, yet challenging experience to the gamer. In theory, this AI would be able to react not only to a player's individual actions but also to patterns in these actions. The end result would be a game in which the player would be able to play against NPCS that are able to adapt and react to their style of play.

As a means to achieve the goals of this AI we sought to create a campaign game which would allow versatility in the possible strategies for a player. Using a campaign mode would allow lengthier battles between the AI and the player. Additionally, this style of game would allow more depth to a player's possible strategies. These features would present more material which could be used to train the AI.

### *Team:*

The Q-BMW team consists of two developers (Emuye Taylor and Brandon Lees). The time frame to complete this project was two months. Based on the limited number of possible man hours given the size of the team and project duration the project goals were limited to a demo version which explored the solution outlined above to static AI. Additionally we sought to modify an existing game with this solution as opposed to creating a game from the ground up.

***Tools:***

To create a game to demo our AI solution we used the Quake 3 open source. Additionally, to create campaign style maps for this demo we used GTKRadiant. Development was conducted using both Linux and Windows which was possible because Quake 3 is designed to be cross platform.

## What Went Wrong:

***Quake 3 Engine as Base:***

For reasons outlined previously we chose to use the Quake 3 engine to create Q-BMW. However, due to the fact that neither group member had any previous experience with this engine, there was a significant learning curve for determining how even the basics of this code worked. Additionally, after we were more familiar with this code we realized that many of the implementation techniques used were not optimal for modification, especially for our purposes. A main problem was the lack of encapsulation of this code. In order to modify one element of this code, it was often necessary to modify many other elements. Additionally, the general design of the Quake 3 project was much more functional than logical. As a result, it was often difficult to determine how to correctly modify certain aspects. Also, because Quake 3 is designed to be a multi-player game player over network the client, and server were completely separate. As a result, sharing new information between these two elements posed a problem.

***Campaign Game Transition:***

One important change which we sought to introduce to Quake 3 to provide for the transition to Q-BMW was to make the game follow a campaign mode. Although, we made some changes to make the game a campaign game, because the focus of our project was the AI, making the full transition proved to be too time consuming. Some examples of campaign game features which we were able to achieve:

- The addition of many AI bots to a level.
    - Prohibiting these AI bots from re-spawning (as would normally occur in Quake 3).
    - Prohibiting these AI bots from targeting or decreasing the health of any bot (which does not include the human player).
- The creating of some campaign style maps which featured larger areas than traditional Quake 3 maps.
- The addition player lives which decrease with the death of a player.
- The addition of a game end based on the player life count, or enemy count.
- The addition of a score system.
- The addition of bot classes which are able to use different weapons, whose deaths increase the score by varying amounts, and acted based on different AI.

Some examples of campaign game features which we were unable to achieve:

- A high level objective to the game.
- The inclusion of transitions between maps.

***Map Making:***

Due to our inexperience with map making and GTKRadiant, creating maps for Q-BMW proved difficult and time consuming. Additionally, because of this time commitment, in order to achieve significant advantages towards a campaign style proved to be out of the scope of this project. Although creating substantial maps would have greatly advanced this project, a team member solely in charge of art, map design/layout may have been necessary.

***Limited Scope of Actions:***

For reasons outlined previously the scope of the project was limited. For example, the number and type of the actions defined for he AI was restricted. The actions were primarily movements centered around the AI's target and cover positions. Additionally, movements were defined to allow the AI to attack and change weapons. Although, these actions allowed for a functional bot which could effectively kill its target, to create more complicated strategies for the AI more complicated actions would need to be defined. Below are several areas which could be expanded to provide such results:

- Movement – The current implementation allowed the bot to move towards and away from cover and its target. Possible additions could include movements which would allow the AI to come from behind its target, or other strategical positions. Additionally, expanding the movement base to include simple movement based on directions might allow the AI to learn to dodge its target's attack.
- Team Based Actions – Currently the AI bot's actions are all independent. A possible addition would be to define actions based on the number of AI bots within a certain proximity. As a result, bots could learn to take action based on the number of teammates that were near them. It is important to note, that for this addition it would be necessary to modify the state representation of the game to include information regarding the AI's teammates.

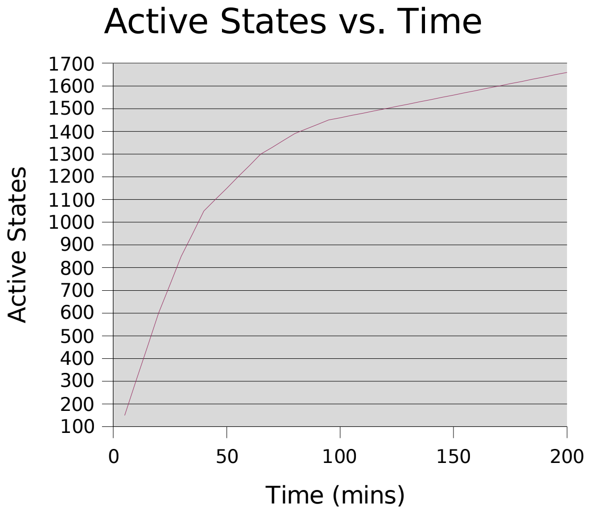
A further result of the limited scope of the AI actions relates to the measurable performance of the AI. Because the AI's actions are so limited it is difficult to determine if the approach used for Q-BMW would allow the AI to learn to play differently based on the different styles of the person playing the game. Although, there were some observable differences in the AI's style, like if a player had a tendency to stay away from the AI it would approach them, and if a player had a tendency to approach the AI, it would use cover more often. However, it is difficult to determine if these differences would be significant enough to truly account for the differences in possible player's styles.

Another addition which could greatly improve the performance of this AI would be to incorporate a sequence bonus to reward sequences of actions which result in the AI achieving its goal. Although a specific implementation plan would require more analysis and planning, if a sequence based reward system could be developed to incorporate into Q-BMW this could add further complexity to the AI's actions. For example, the AI could learn sequences of actions like dodging in combination with shooting.

***Peak in State space Exploration:***

Through analysis of the active states in several trials of Q-BMW it became apparent that the number of explored states has a tendency to peak around 1500 (see **Figure 1**). From 0 to 1500 the number of active states increased rapidly. However, after about 1500 states had been explored, the number of active states would increase at a much slower rate. This peak may be related to a peak in the AI's performance. More specifically, between 0 and 1500 explored states there is much more observable difference in the AI's performance. After 1500 states, although the AI does show improvement, this improvement is much harder to see, especially for a player less experienced with the project. As a result of this peak in the state space exploration, it would take many hours of game play to explore all of the states. Until all of these states have been explored, there would be many states where the AI still took the default random actions. In order for the AI to have a completely optimal policy, and show optimal performance, the AI would need to not take any random actions, and the current peak in the AI state space would have to be corrected. This characteristic of the state space has many possible explanations. One such example is that the state representation of the game includes combinations of game features (AI/player health levels, AI weapon, AI position relative to target, etc.) which are not often realized. A possible solution could be to keep less detailed information regarding some features, and more detailed information regarding others. Although, determining the specifics of these detail combinations would take much more analysis.

**Figure 1.**



## What Went Right:

***The AI Actually Learned:***

The AI showed noticeable differences in a very short amount of time. Trials of Q-BMW were conducted with gamers of various levels of first person shooter experience. Regardless of this level of experience noticeable differences in the AI's play was observed. Some examples of AI learned behavior include:

- The AI quickly learned to avoid being shot. While the AI was still taking its default random actions, it would allow you to shoot at it. However, after only 50 explored states (see **Figure 1.**), it would run or try to take cover to avoid being shot.
- The AI learned to attack when appropriate. When the AI was taking random actions there was no consistency to when it tried to attack a player. However, after only about 15 explored states, the AI would fairly consistently shoot when approaching, and rarely shoot without a target.

To demonstrate our AI we prepared two demo versions of Q-BMW. The first version featured AI which did not have previously learning. The second version featured AI which had ongoing learning (of between 2 and 4 hours). Many people, with various levels of first person shooter experience played against the AI. These trials provided evidence that the AI without previous learning was consistently easier to beat than the AI with previous learning. Figure 2 below illustrates this evidence.

**Figure 2.**

| ***Version 1 (without previous learning)*** | ***Version 2 (with previous learning)*** |
|---|---|
| 3800 | 1400 |
| 5600 | 300 |
| 2400 | 2500 |
| 3700 | 1500 |
| 1400 | 5600 |
| 5600 | 1300 |
| 3200 | 1400 |
| 5600 | 2400 |
| 5600 | 3700 |
| 2800 | 2500 |
| 1200 | 1000 |
| 5500 | 1700 |
|  |  |
| **Average: 3866** | **Average: 2108** |

***Q-BMW is Easy to Play and Fun:***

Due to the popularity of first person shooters in the gaming community, most players at the Q-BMW demo had previous experience with this type of game. As a result, it was very easy for them to sit down and begin to play without any instruction or explanation. This small learning curve allowed us to demonstrate the innovation of our learning AI clearly without isolating people who already enjoyed this type of game. Furthermore, people who were unfamiliar with the techniques used to develop the AI were able to see the AI change through the game and understand what we had accomplished. Many of the users at the demo commented that the game was fun and many people played the game many times through, often coming back several times during the demo. Additionally, since we used Quake 3 as the base for our project, many people commented on the noticeable differences between our AI and the Quake 3 AI. A common comment was that the AI was less scripted than Quake. This indicates that we achieved our goal in creating a less static AI.

***Component Modulization Allowed Fast Development:***

When planning our project, we focused on separating all development into well defined parts. Some examples of these components include the q-learning implementation, AI action definition, AI action convergence, and modifying Quake to create a campaign game. Our approach was to develop and test these components independently. This allowed us to focus our efforts on the current task. Furthermore, by incrementally testing the components as we developed them, we could be confident that past code was fully functional allowing us to focus on moving forward.

Additionally, because the majority of the AI code was developed as an independent module, most of it is not reliant on Quake. As a result, it could be easily adapted into a different game. This would be beneficial if work on the project were continued in the future since Quake 3 is somewhat difficult to work with and may not have been the best choice for a campaign game.

***Project Completed According to Schedule:***

A common problem with software development is difficulty in foreseeing potential problems and creating an accurate schedule. By thoroughly planning our project from the start, we were able to develop a schedule that accomplished our goals and was possible to follow. Although we did encounter problems throughout the course of the project, we had no major setbacks and were able to meet our weekly deadlines. The largest setback we encountered was avoiding giving in to our own enthusiasm for the project. While working on the project we were constantly thinking of new ideas and modifications that would improve the project, however due to the limited time frame we were forced to scrap many of these ideas. Although this was a setback to the project, it did not have a negative impact on completion of the project and was a much more positive aspect than the usual expected setbacks in software development.

***Open Source Allowed For a Completed Game:***

The decision to use an existing open source game as the base for our innovation turned out to be a good idea. While our project was similar in scope to the other projects in the class, we were able to deliver a functional game in addition to being able to focus on our innovation. This seemed to be a problem for many other groups who focused their efforts on innovation and had difficulty creating a game to effectively demonstrate it.

## Conclusion:

Although, throughout the course of the Q-BMW project there were setbacks and limitations, the project overall was a success. The high level goals set forth in the beginning of this project were met. We were able to develop an AI that is able to learn and change it's behavior based on actions of a human player. This AI is in many ways more dynamic than that of existing first person shooter AI , even though the scope of the project was limited by the available development time and the size of the development team. Although these limitations prevented this project from being  something which could, in its existing state, be put into current games, the success of this project illustrates the feasibility of our approach. Analysis of the results of this project demonstrate that there are many areas that could have been improved, many of which we discussed previously in this document. Regardless of these improvements, it is apparent our approach could be scaled into an AI for a commercial person shooter.