Coursework 2: Understanding Social Behaviour in Games

Thomas AE. Smith

April 18, 2014

1 Introduction

This report describes the adaptation of a POSH agent for the Unreal Tournament 2004 game, in order to add additional capabilities that make it more effective in a capture-the-flag tournament environment. This involved developing code to specify a new behaviour, and modifying a provided basic plan in order to support the new behaviour and a range of new competences. The overall task was akin to a set of mid-development iterations of the BOD methodology.

2 Approach

The first modification required by the specification was to add jumping capability to the agent. The provided agent was unable to jump over objects in its path, however the spawn point in the tournament map was gated by a low obstacle. Adding the new capability consisted of three tasks - providing code for the behaviour's actions and senses, adding these actions and senses to suitable locations within the agent's plan, and testing and debugging the implementation.

2.1 Behaviours

A new behaviour was developed, in order to provide the senses and actions required to support jumping over obstacles. Ideally from a design cohesiveness standpoint this functionality would have been added to the existing Movement.cs file, however the provided code could not be modified, and so the new code resides in a separate behaviour.

It transpired that the existing `IsStuck()` sense provided by Movement.cs was generally sufficient to indicate suitable trigger points for executing a jump action, and so the only code added was an action that sends a JUMP message.

2.2 Plan

In order to make use of the new action, a new drive element was added to the top of the drive collection — i.e. with greater priority than any other element. Following this approach ensured that whenever the agent was moving for whatever reason, if it became stuck it would attempt to jump over whatever obstacle was blocking it. An alternative approach would have been to add the check and response to the `moveto-selected-nav` competence, however if alternate movement competences were to be added in the future, they would also need to duplicate the check and response.

`jump-obstacle`: If `IsStuck()` = 1.0, then follow the `jump-over` action pattern: send a JUMP message and reinitiate movement.

Aside from the ability to jump, a number of other competencies are needed for an effect tournament agent. The provided agent did not engage in combat and made no effort to retrieve its own flag if it was captured by the other team, and so elements were added to the plan in order to support these tasks. In decreasing order of priority:

`return-our-flag`: If our own flag is on the ground (it has been captured but the carrier was killed) return it to our own base in order to avoid having it recaptured. Select it as our navigation target, and move to it in order to return it.

`shoot-enemy-carrier`: If our own flag has been captured, and we can see an enemy carrying it, shoot at that enemy. This element uses a provided sense and action from the Combat.cs class, and has a higher priority than capturing the enemy flag — as without our own flag present at the base we cannot score.

`attack`: If we can see an enemy, and we have both a gun and ammo, then select that enemy as a combat target and shoot at it for as long as we can see it. This has a lower priority than capturing the enemy flag, which allows for opportunistic attacks, but avoids pursuing enemies to the detriment of other goals.

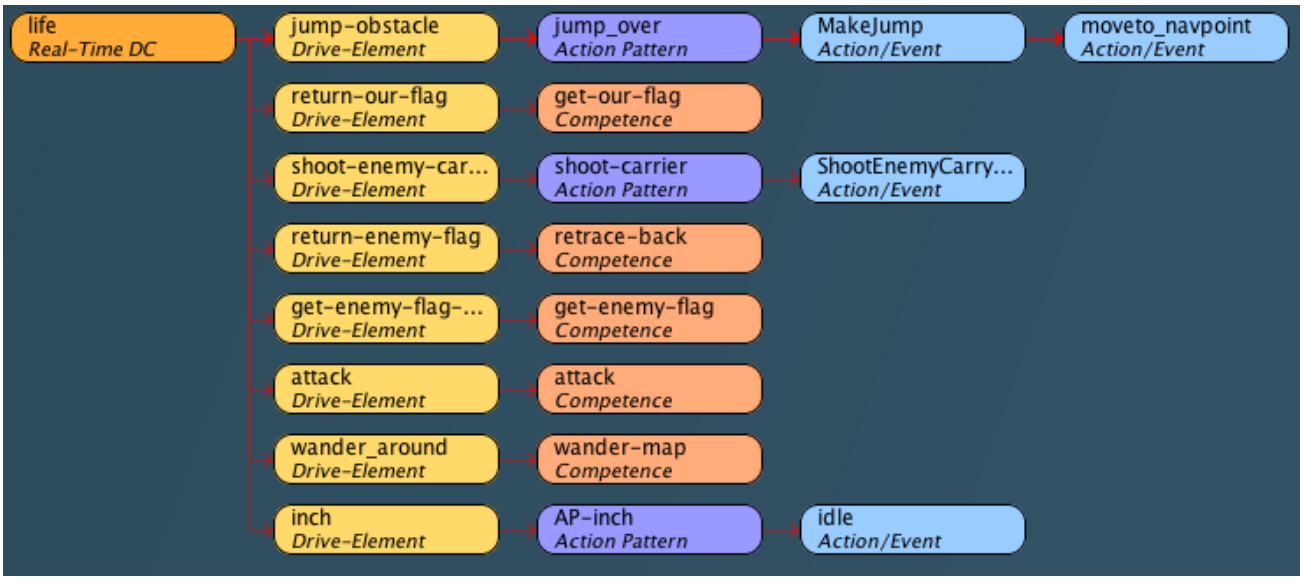

Figure 1: Overview of POSH plan

3 Results

Basic testing indicated that the modified agents performed better than the provided agents. On the provided testing map (`CTF-Bath-CW3-comp.ut2`), the low obstacle gating the spawn point completely blocked the original agents, which were unable to jump over it. On a version without the obstacles (`CTF-Bath-CW3-small.ut2`), the modified agents had an advantage due to the occasional opportunistic attacks against the non-violent provided agents. The provided agents also occasionally exhibited a bug where once killed they would thereafter not navigate properly — this was fixed in the modified agents by removing references to `focusing_task` from the `wander_around` competence, and adding a `selected_navpoint_reachable` check.

4 Discussion

The implemented additions to the behaviours and plan of the agent made it more suited to the capture-the-flag tournament environment — however a number of other functionalities would also have been possible. The API supports the ability to be aware of and respond to incoming projectiles, or more advanced improvements such as improved pathfinding, varying agent roles or even inter-agent communication would potentially have improved the agents' overall performance.

5 Conclusion

The modified POSH agents were more effective in a capture-the-flag tournament environment than the provided POSH agents — in some situations considerably so due to their ability to jump. The BOD methodology guided the development of iterative improvements to the agents' behaviours and plans, and made it easier to upgrade the provided code.

A Plan Modifications

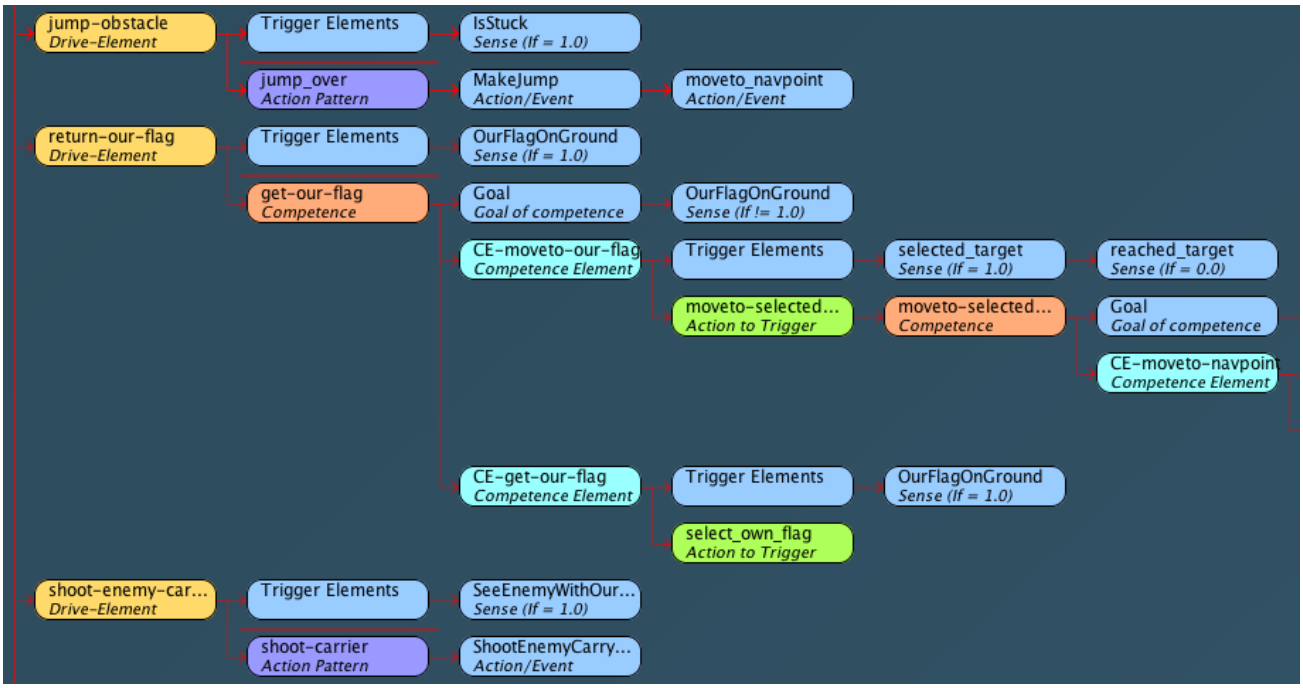

Figure 2: Jump and flag return behaviours

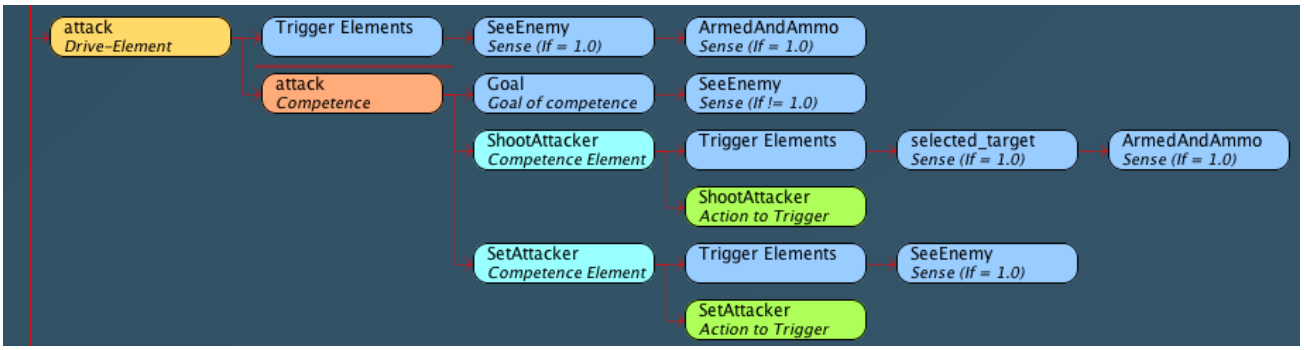

Figure 3: Attack-on-sight behaviour