Unreal Tournament 2004 - GameBot Observations

Dominic Hauton

April 7, 2017

Drive Fall Through

Detecting if the bot is stuck wastes an action, so the bot jumps on a regular basis regardless being stuck, which is wasteful but necessary for fast navigation. If some drives were allowed to fall through, this could be represented as a drive instead.

Stopping Actions

After combat un-crouching and stopping shooting is difficult in BOD as every event that does not involve combat has to request this in the plan. A better method discovered was using an action timeout, that sent the ending command a few seconds after the plan completed as part of the initial behaviour.

Trigger Inheritance

The provided software tools do not provide any way for drives and competencies to inherit triggers from behaviours. This would be useful to prevent user error, an example being the requirement for actions that shoot to check if the bot has ammo as a trigger.

BOD Plan Checking

In the tools provided there is no way to check that there is always an action being performed. A tool to check if one of the drives will be satisfied in every case would be useful, similar checking that code always returns a value in a function.

Plan Execution Speed

The BOD framework does not run at a consistent speed between simulations, especially when in Debug mode. This dramatically affects bot performance, as bots on a faster loop time are able to react much faster to events, demonstrating that planning rate can impact bot performance.

Downsides of Culture

Runner bots with the same goals acted similarly when faced with danger. This was detrimental in some cases, for example bots standing in the way of other bots while shooting, which severely impacted combat performance in groups.

Benefits of Culture

By using three runner bots concurrently complete their goals, a much higher rate of flag retrieval is achieved. When two of the bots die the goal can still be achieved by the third bot. This is tested by using varying proportions of defender and runner bots.

Latches

Using latches enables behaviour persistence throughout multiple BOD cycles. When a bot starts shooting and crouching for combat, it no longer has to stop shooting and stand up between cycles. This leads to more consistent and accurate combat.

Plan Simplification

Simplifying the plan makes the bot behaviour more predictable and easier to debug. A cycle of adding drives to the plan until the desired behaviour is observed, proceeded by making those drives as simple as possible made bot behaviour easy to modify throughout development.

A* Path Finding

Adding A* path finding allows the bots to navigate faster to their destination as the bots no longer need to meander to their destination. Navigation to the enemy flag from the home base with path finding enabled is more than halved.