An Introduction to AI in Battlefield1942™

Author: Tobias Karlsson

Abstract: This document will try to give a brief introduction to how the AI of Battlefield1942 works. It is by no means exhaustive and is only intended as a support for people wanting to experiment with the AI.

This is the first document in of a number of documents that aims to describe how to use the AI of Battlefield1942.

Understanding the AI

This chapter will give an introduction to how the bots work. It is crucial to have a basic understanding of the inner workings of the bots in order to use them correctly. The intention of this chapter is not that a reader should be able to modify any code, just to have a basic understanding of how why the bots behave the way they do.

Reactive Agents

The bots are modelled as reactive agents. This means that they are that they are programmed using the principles of agent based programming and reactive AI. This has important and far reaching implications for the resulting behaviour of the bots.

Agent Based Programming and the Bots

The researchers of the field has not been able to agree on an exact definition of an agent, in spite of this obvious weakness, the approach has many interesting benefits. Despite the fuzzy, or lacking definition of an agent, it is possible to agree on a number of different properties that an agent must have. Among these properties are that the agent must be able to accomplish its task in an dynamic and complex environment that the agents lack both full control over and full knowledge of. The agent based approach to the bots in Battlefield1942 results in a number of interesting properties; the most significant is the independence of the bots. The bots are quite capable of managing themselves without any guidance of humans once the game has started. It also means that it is possible to create a bot when the game has started and the bot is able to participate in the game without having participated in any pre-game start-up phase.

As agents, the bots is also able to handle new, altered, and conflicting goals. This means that the bot can take orders from both other AI entities and human players, weight them against their own needs and chose an line of actions that it think will best meet the different requirements. In effect, it is the bots that have the final call in deciding what to do, and they are free to disregard any orders given to them.

The bots are also responsible to gather their own information about their world and they do so using imperfect and local sensors. This means that it is possible for the bots to miss judge or totally miss some of the traits of the world in their surrounding.

Reactive AI

Almost any AI program can be characterized as either planning or reactive. A planning system spends some time producing a plan that it then tries to execute. A planning system needs a good understanding of its world and to be able to foresee at least the most significant results of its actions. These systems are generally CPU intensive and also have problems when the world develops in a way that is inconsistent with their plans.

Reactive systems work in a fundamentally different way. They only react to changes in the world, and they do so by having a more or less simple direct mapping from sensors to effectors. It does not prevent the AI from having long-term behaviours, but it does require that the agents change either internal state or the state in the world so that they can detect in which part of the behaviour they are in. A reactive agent cannot foresee any the actions of another agent, it can at best react to its actions.

The bots of Battlefield1942 are reactive. This has great implication on how they behave and what they can do. It is important to realise the limitations of reactiveness. It is not possible to use the bots for long term planning, that is, any attempts of making the bots perform deliberate and proactive planning will most likely fail.

As the bots only react to their environment, they can take appropriate actions at the very moment they spawn into the game. Even though they use a small amount of memory to keep track of the development of a number of different variables of their world, they can perform with any of these variables set to the default value.

Basic Concepts Specific to Battlefield1942

The AI of Battlefield1942 is divided into three main systems, which will be described briefly below, and in more detail later on in this document:

- Bots
- SAI (Strategic AI)
- Support systems

The SAI and the bots are responsible for waging the war together. Each bot is a soldier that act autonomously on the battlefield, their responsibility is to try to keep their soldier alive and at the same time killing as many enemy soldiers as possible. The SAI's responsibility is to coordinate the efforts of the bots so that they all attack and defend the same positions.

In the category "Support systems" are a number of crucial systems, most of them concerned with answering questions about the world for the bots.

The Bots

Each bot consists of a number of behaviours that compete for the control of the bots. Exactly which behaviours that are present is decided by the vehicle the bot is operating at the moment. This makes it possible to make the bot dynamically switch between a variety of different vehicles without needing to hardcode any information about any of them in the bots implementation. To make the system yet more flexible, the behaviours are divided into three different parts that can independently be changed so that the code becomes more reusable. These parts are called the *behaviour* that is responsible to calculate the need of performing the behaviour at the time. The *plan generator* generates a static short-term plan in the *BAP-language*, a generic language for the bots of Battlefield1942. The last part is the *plan interpreters*; their task is to interpret the commands of the BAP-language that the plan generator generated, and to execute them in a timely fashion.

The SAI

The SAI is a partly distributed reactive system. It depends on the entities called *Strategic Areas* to calculate and provide it with most of the information it needs. The strategic areas are responsible for calculating a temperature that says how important the area is, and also a need of resources. The SAI is then responsible to mediate between the needs of the different strategic areas, and to distribute resource between them. The strategic areas and the SAI cooperate in moving and commanding the troops.

The Support Systems

There are, as mentioned, a number of different support systems that the AI use. Below, there will be a brief overview of the most important systems:

The knowledge system is a system that is a layer between the AI and the rest of the world, it wraps around the object system. It is comprised of three main parts. The first part is the *AI Object Templates* that hold generic and constant information of the different object types. The templates hold information that have been manually written into a con-file as well as information that have been extracted from the object in run time. The *AI Objects* are instances that are attached to each object that the AI is interested in, they map directly to both the real object and the AI Object's template. The *Information* is a class that is connected to an AI Object. The information does not mirror the AI Object, but it holds what one side knows about that object. It is possible for the Information object to differ quite a lot from its corresponding AI Object.

The sensing system is responsible for the "visual" input of the bots. The system casts rays in the world against other objects in order to decide if they are visible for the bot. The sensing system is one of the more CPU intensive systems of the AI.

The path finding system is responsible to produce paths that the bots can follow through the world. It uses a hierarchical A* on a 2D bitmap. One important limitation of this system is that since 2D bitmaps are used, the bots cannot move between places directly above one another (i.e. floors in multi story buildings) because only one pixel is used to represent each position in the world.

Real Time and Time Slicing

Battlefield1942 is a real time system, which means that all subsystems have a very limited amount of time to perform their calculations on. This is a very significant constraint on the system, and also the AI has to abide to it. The AI system is time sliced, that means that all the major systems can abort, break, or postpone their calculations if they run out of time. There will frequently be situations where the AI cannot perform all the calculations that the system would need to perform optimally. This can result in a number of typical failures (ordered by severity, worst case last):

- Stalled path finding.
- Failed sensing.
- Inadequate plan selection.
- Lack of a plan.
- Performing of dated actions.
- No action due to stalled action updating.

It is important to recognise these symptoms, as it is easy to think that the problem lies within the AI, and not in the fact that there is a lack of time. All symptoms will be described in detail below:

Stalled Path Finding

There are only a limited number of path finding searches that can be performed at any given time, also longer path findings may take several frames to complete. If this occurs and there are many bots that need path finding at the same time, some unlucky bots may have to perform without path finding for some time. For obvious reasons, bots without path finding will not and cannot move.

Failed Sensing

The sensing system is also sensitive to low CPU quotas. If there is little time available, fewer bots will be able to perform sensing at a time, and also the quality may be affected. Limited time for sensing means that there might not be time to sense all nearby objects and that fewer rays may be cast against those objects sensed, thereby increasing the risk of not spotting the objects. Bots that are pressed in their sensing may completely ignore enemies.

Inadequate Plan Selection

If the bots lack time, it is possible that they may not consider all possible behaviours. More critical behaviours, like dodging collisions, and attacking enemies are more likely to be considered than less critical behaviours like changing vehicles or moving towards control points. If a bot lacks time to consider all behaviours, it may still have chosen the optimal one, also, bots vary their behaviour somewhat, and if it is not very obviously which behaviour that is the optimal one, it may chose another behaviour. If bots misses to jump into vehicles, that may be a sign of lack of time.

Lack of a Plan

The bots regularly finish their plans, either by reaching ends of the plans, or by the plans somehow becoming invalid. If the system is short on time, the bots may have to wait before they are allowed to update their plans. With no plan, the bots are completely helpless and will not be able to do anything. As noted above, bots frequently run out of their plans, this is normal behaviour; the problem arises when they do not get a new plan.

Performing of Dated Actions

If the system is really pressed, some of the bots may not be able to update their actions in the plan, in that case the bots will keep its current input, that is, they will not revaluate their action but just keep on giving the same input as the last frame.

No Action Due to Stalled Action Updating

In extreme cases, when the bots are completely out of time, and are not allowed to update their actions for a number of consecutive frames, their input will be reset, and they will do nothing. If this happens, the whole system is probably seriously overloaded.