BamBirds 2017

Thu San Cao, Lukas Lengenfelder, Ute Schmid,
Kevin Volkert, Katharina Rupp, Anne Schwarz,
Andreas Rayzik, Felix Dräßel, Moritz Lintner
and Diedrich Wolter

Approach

The BamBirds agent implements a classic AI approach using a rule-based system that operates on an abstract scene description. The rule-based system essentially implements an "affordance-based" approach to tackle a level in the game: the agent aims to exploit modes of destruction *offered* by a configuration in the scene in a greedy fashion. If there are complex and fragile buildings, the agent aims to destroy them by taking out a supporting element; if there are high-rising tower structures near goal objects, the agent aims to make them collapse towards the goal object; if there is TNT in the scene, the agent aims to make it explode; and so forth. For each possible application of the rule, the rule-based system determines a likelihood score that captures certainty of success (e.g., likelihood of a structure to collapse when hitting a specific object within the structure) and the expected benefit of performing the shot. This part of our agent is based on an abstract description of the level, using concepts such as structure (a connected set of game elements typically forming shelters, etc.) and spatial relations such as above, in front of, or behind.

Overall control of the agent is performed by a meta-cognition module. One task of meta-cognition is the final selection of the shot, using candidates and scores determined by the rule-based system. Since the rule-based approach can systematically fail to recognize good/bad shots, shot selection is partly based on scores determined, on previous performance, and on random. Once every level has been tried, we choose a level with a probability according to the estimated increase in the overall score possible, i.e., scores achieved by opponents are not considered. The estimation is countered with the number of attempts already spent on that level, so expectations get lower if we tried over an over.

Agent Realization

Implementation of our agent is partly based on the naive agent provided. Components for image processing, client-server interfacing, and shot computation are used by the BamBird agent. Aside these components provided as Java code, the BamBirds agent comprises the meta-cognition module and a scene description module which transforms a list of objects obtained from the vision module into a qualitative symbolic scene description. Scene descriptions are output to a file using Prolog syntax. Our agent invokes Prolog to determine objects that are promising to shoot at. Determination of shots is done in a rule-based manner, rules are written in Prolog.

Our 2017 agent is based on a completely rewritten rule base as compared to our 2016 agent. Also, the 2017 agent encompasses an advanced meta-cognition module. The remains of the code are taken from our 2016 agent, but we have improved quality of the code.