

# Report of Team BanzAI - Robocup Rescue 2003

Silvain van Weers    Christian Vossers    Stefan Leijnen    Andrew Koster

10 september 2003

# Inhoudsopgave

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>The program</b>	<b>4</b>
2.1	The Framework . . . . .	4
2.2	The A.I. . . . .	4
2.2.1	The plan . . . . .	4
2.2.2	The Reality . . . . .	5
<b>3</b>	<b>Building the BanzAI Robocup Rescue Team: Progress</b>	<b>5</b>
3.1	March - July 2002: First attempt . . . . .	5
3.2	September - December 2002: Setup . . . . .	5
3.3	January - March 2003: Development . . . . .	6
3.4	April - June 2003: Preparation for the WC . . . . .	6
3.5	July 2003: World Cup in Padova . . . . .	6
<b>4</b>	<b>The World Championship</b>	<b>7</b>
4.1	Comparison to other teams . . . . .	7
<b>5</b>	<b>Future work</b>	<b>7</b>
5.1	FireAgent behaviour . . . . .	7
5.2	PoliceAgent behaviour . . . . .	8
5.3	AmbulanceAgent behaviour . . . . .	8
5.4	Pathplanning . . . . .	8
5.5	Communication . . . . .	8
5.6	Robocup 2004 . . . . .	9
<b>6</b>	<b>Conclusion</b>	<b>10</b>

# 1 Introduction

This document is a report about the participation of Team BanzAI at the World Cup Robocup Rescue in Padova and the project that preceded the participation. The Team BanzAI source is freely available and this document also describes its structure and use. We have an evaluation of our agents as compared to other agents and the final chapter is about what work is left for the future and possible participation at the World Cup 2004 in Portugal.

## 2 The program

### 2.1 The Framework

Our Robocup agents were built from scratch, making no use of the ADK or other existing frameworks for Robocup Rescue Agents. This means we had the freedom to redesign the architecture the way we felt would work best. The bare framework is functioning well. We have the main class, which contains the main method. It starts the program and parses possible parameters for further use. It then gives the controller the assignment to start all the agents and connect them to the kernel. All agents started by a process are run in the same thread. The superclass agent is the `AbstractAgent`. All agents are extensions of it. The `ConcreteAgents` are implementations of the lower class abstract agents, so there is a concrete firebrigade, ambulance, policeforce and one of each of the centers. For more information on how to make agents for the Robocup Rescue Simulation, we refer you to the documents [3] and [4].

The agent's actions are decided in the `act()` method. The `hear()` method receives the messages from the kernel. Apart from the `act()` method there are other classes that help in the process of deciding the command to be sent. We use a pathfinding algorithm to search for a path and the `TargetDetermineSystem` class is used to find a suitable target to either move to, extinguish, clear or rescue.

The agents are started by running the executable `banzai`. Possible parameters are `-A 1`, `-P 1` or `-F 1`, which each start up only the Ambulances, Policeforce or Firebrigades respectively. If these parameters are used one can still start the program again with no parameters to connect the remaining agents.

### 2.2 The A.I.

#### 2.2.1 The plan

The original idea of how we wanted our agents to behave was laid out in our Team Description Paper. A short summary follows. For the complete ideas see our 4-page Team Description Paper.

Individual agents try to behave optimally to solve the problem for their own situation. This is based on making decisions from their own local sense or hear information. On the other hand, the autonomous behaviour is based on a so called utility-model, which is overall based on global information. Each agent holds an abstract version of the map where additional information is stored. A utility value for each of the objects in the abstract map is calculated, indicating its importance. The utility is derived from information that is known about the object. With the use of a complex path algorithm, that uses  $A^*$ , Dijkstra's shortest path algorithm and a self-made algorithm called  $V^*$ , an agent tries to move to the target which seems most important. Agents also communicate to centers about the targets to update crisis information. Centers, in their turn, decide which agent needs to know this additional information. Also, agents may do requests over the communication channel when the current situation needs assistance. The center then decides if extra agents are to be assigned to the target of the requesting agent. A center may overrule autonomous behaviour of an agent at any moment, when the center sees fit. This decision becomes critical in the dispatch problem. As a result, centers will distribute agents over the field, and assign them with new targets. By optimizing the global knowledge of agents by means of communicating utilities and by extending the autonomous behaviour, agents have a good representation of the overall crisis scenario. The use of a utility-model and a set of direct pathplanning algorithms establishes efficiency in a satisfying way. Developing agents that perform well, even in a worst case scenario is, in our opinion, the key issue of the RoboCup Rescue Competition.

### 2.2.2 The Reality

Due to a shortage of time most of these ideas were not implemented in the final version. We implemented some easy methods of sending firebrigades to fires and ambulances to injured humans. The police force agents did work as planned, although these too could have used some improving. What happened in the final version is that all firebrigades were sent to the same fire by the center. When that was extinguished they would all be sent to the next fire. This system was always used for each agent, regardless of whether a firebrigade was able to move there or not. The method for choosing the fire was also rather crude, because there was no algorithm present to discern whether a fire was at the edge of a cluster of fires or not. The ambulances were largely autonomous, they would drive about the city looking for injured civilians and rescue these when found.

## 3 Building the BanzAI Robocup Rescue Team: Progress

It took us over 16 months to build the rescue team that was used at the WC in Padova. This time can be separated in five different periods, each representing a different fase of the project:

- March - July 2002: First attempt
- September - December 2002: Setup
- January - March 2003: Development
- April - June 2003: Preparation for the WC
- July 2003: World Cup in Padova

In this section, each of these five fases will we be explained. The problems we encountered are explained, as well as the (possible) solutions. The description of the final fase will mainly be a summary of the events that took place in Padova. After reading this overview, the reader should have a clear picture of the progress of our team throughout the year.

### 3.1 March - July 2002: First attempt

During the last two periods of our second year, all four of us - being part of a much larger group of 13 people - participated in the Softwareproject, where we were instructed to build a team of agents for the robocup competition. We started out well, by setting out timelines and dividing the group into three subdivisions (Architecture, AI and Communications) , but after several weeks, problems began to arise. We had decided not to use the Agent Developers Kit, but to build our own version of it instead. However, this had cost us much more time than we anticipated. Actually, it had cost us about four months (and even then it wasn't quite stable). Because the AI group had to wait for it to finish, hardly any of their ideas could be implemented. The project was not really a success, although we had managed to build working agents, which was hard enough on itself. But the AI of our program turned out to be a deception, especially when one looks at the prolific plans that were de veloped on paper.

### 3.2 September - December 2002: Setup

After a holiday of two months, most of us were still talking about the project. Many participants had still been thinking of what we should have done better, and how. The four of us finally decided we wanted to take another shot at developing the agents we really wanted: inspired by

the previous framework, but with a more stable base, making use of all the AI techniques we devised earlier.

We set out to gain support for our project, and most people were very helpful and enthusiastic. A number of articles were selected for us to read by Marco Wiering, who also gave us several lectures. All this gave us inspiration to further develop our own ideas. The Robolab had some space left for us to work in, but it was hard to find computers to work on. The simulation would only run on a certain version of Linux, which was not available in the regular working areas. This meant we had no place to start implementing, until we could work in the Robolab.

### **3.3 January - March 2003: Development**

In January, almost two months later than we had originally planned, the implementing could finally start. Although we still only had 2 computers. If there was one thing we could change in the course of the project, it would be installing four workstations in September, so we could've started building the framework sooner and had the time period between January and March for implementing and testing our AI ideas. Now we were forced to take some "extreme programming" measures: one person implementing, the other looking for errors and maintaining an overview. After working like this for nearly two months, our agents were doing much better than those of the Softwareproject version.

But then we suddenly realised that the WC admission deadline was nearing. We not only had to hand in a log file, but also two papers describing the features of our agents. This was a very busy period, during which we ran many simulations in order to obtain a nice logfile, and worked hard to write the two papers. After handing them in, we took a break for about two weeks, until the submission result was posted. All our work had not been in vain, the result stated we were through.

### **3.4 April - June 2003: Preparation for the WC**

We now faced another problem: most of us had to do an internship in May and June, which required much of our time. We still did some work after office hours, but there wasn't much progress. Besides, preparations had to be made for our stay in Italy: an apartment had to be arranged, as well as sponsoring, grants and publicity. It was difficult to see other teams working hard to put the finishing touches to their code, while we were forced to work on other projects. But at least we managed to get proper funding for our trip.

### **3.5 July 2003: World Cup in Padova**

Although we might have treated our stay in Padova as a holiday, we decided to work on our agents instead, to make up for the time we had lost in the previous two months. While the sun was shining outside, we were in the hall, finishing the agents. Another thing we would change if we could: making sure the agents are finished before you leave, so you can fully concentrate on the matches and other events taking place around the conference. After several days (and nights!) of programming, it was time for our first match. Things did not work out as planned, somehow the agents performed worse than in our previous runs, and definitely worse than the other teams. Unfortunately, we finished last. But we did not leave as losers, because we had learned a lot.

## 4 The World Championship

As mentioned above, our agents didn't have the desired functionality that we had planned on implementing from the start. It was therefore no surprise to us when our agents didn't work as well as other teams at the World Championship. We were expecting to do better than we actually did, but can point out several reasons for coming last.

### 4.1 Comparison to other teams

It is not quite fair to compare our agents to other teams' agents. Most teams have more experience with the Rescue league than we do and also more knowledge of Agent Technologies. We weren't expecting to be able to compete with the top teams present, but expected to be able to compete with the lower regions of the tournament. Even so, we ended last, but on score only, compared to some of the last 4 teams our agents were not really much worse. For example, in one match, a team of agents crashed on startup, and yet we ended below them. The reason for this was the failing of our ambulances to rescue civilians. The roads got cleared by the police agents and the civilians were therefore enabled to run into buildings. There they were far less safe than blocked up on the street, because buildings can catch fire and a lot of civilians died in this manner. Our score was lower than the one of the crashed agents. We obviously didn't have much chance at competing with most teams, because we lacked advanced dynamic path planning and recognition of the edges of a cluster of fires. Apart from this the teams were not formed intelligently according to location and reachability of a target and the communication between agents was rudimentary at best. Still our agents functioned, were stable and robust and we are of the opinion that this was the best performance possible in our circumstances.

## 5 Future work

The following ideas were seen and read at Robocup 2003 and deserve attention in future development:

### 5.1 FireAgent behaviour

**Coalitions** Intelligent group-forming and reforming with low computational cost by center agents, thus making groups with sufficient capacity for their aquired task.

**Target auctioning** By bidding of agents or groups of agents at targets, controlled by the center agent. Bids are influenced by target and agent: the target its strategic positioning (center of city or at the boundary and number of buildings around them), and the agent its actual position and its group members status.

**Extinguish forecasting** A critical consideration when an agents selects a target is the extinguish prediction. How much time will it take for an agent or agent group to extinguish a fire? An agent its prediction as accurate as possible with limited computational complexity. Extinguish-time is influenced by the size, number of floors and fieryness of a building etc. Since this must be done with low complexity, this can be done with use of Q-learning as in [5]

**Boundary recognition** intelligent recognition of the front of fire-ignition, to predict the spreading of fires. It is crucial to predict which fire to extinguish first [6].

## 5.2 PoliceAgent behaviour

**Recognition** of critical regions: Roads around fires and stucked civilians are more likely to be travelled, and deserve a higher priority. The recognition of these critical regions with higher priority cannot be optimal since this would be too slow.

**Relative road** importance: The likeliness of a road to be travelled and thus have a higher priority to be cleaned can be computed off-line, eg. when the simulation hasn't started yet. By pathplanning between all nodes on the map and afterwards counting the use of each road, a value to each road can be given; a higher use will result in a higher priority. This can be done efficiently with a Floyd-Warshall-algorithm.

## 5.3 AmbulanceAgent behaviour

**Buriedness forecasting** as extinguish forecasting, the likeliness that a civilian will survive during the time that it is being unburied by an AmbulanceAgent must be computed as accurate as possible. If a civilian will not survive during the rescue-action, the rescue of this civilian should not be taken into account, which makes deciding for an AmbulanceAgent less complex. Prediction can also be learnt using Q-learning.

## 5.4 Pathplanning

**Super-roads** the general term for use of relative road importance (see PoliceAgent) in pathplanning. Roads with an higher priority to be cleaned and much lanes tend to be cleared and without traffic jams and thus have an advantage in path-selection. These roads are declared 'superroads' or sometimes 'highways' to describe their special status. Selection of the superroads is also done offline, as in [7], which also take into account many roads to refuges to be superroads (computed offline), and the roads to fires (computed online).

**Exploratory planning** some participating teams used two path-planning algorithms with different knowledge of the environment: the so-called Safe planning (using roads that are known to be unblocked or blocked) and Exploratory planning (which may also contain roads of which is unknown whether they are blocked). So a road can have 3 different kinds of status: unblocked, blocked or unknown.

**Exploration using quadrants** at the initial exploration stage all agents get their own part of the city to explore and find potential targets. This causes agents to be far from each other and reduces the likeliness to explore the same, which is non-optimal.

**Point-based agents** some teams stay using quadrants for their agent the whole simulation. Groups are assigned to a certain part of the city as successfully used by [8]. This reduces the complexity of path-planning and group formation and management, because only a small part of the city needs to be taken in consideration. Agents can also be more up-to-date because of the smaller area.

## 5.5 Communication

**Message priorities** use of a value which indicates the importance of the message that is received. The value is influenced by the accuracy (age) of the used information (older is less accurate) and/or the hurry of the task.

**Message importance prediction** some teams use Q-learning for the prediction of message importance, by evaluating which message tend to have the greatest effect: these messages should be listened to first.

**Platoon Agents** rarely used as a central pathplanning point, platoon agents are mostly used for the correct distribution of information to its team members. Note that a platoon agent does not have to be 'fysical' in range.

**Low communication team formation** The bandwidth for communication is low, so the formation of groups should be done effectivly, with minimal use of messages.

**Communication Lag** Centers need to know all the knowledge collected by agents to make good decisions. The knowledge of all agents together and all centers together can be compared to see how much information can not be shared because of communication bandwidth. The delay, called lag, can also be taken into account as by [9]. This gap and lag tends to be bigger in the beginning and middle of the simulation and gets smaller at the final stage, and is a usefull indication of communication-model effectiveness.

## 5.6 Robocup 2004

Our goal is to have a Team BanzAI II ready for participation at the Robocup Rescue competition in 2004. We will only compete if most of the ideas we want to use are implemented and we feel we can do a lot better than in 2003. The competition was great to participate at, but if we are to compete again, our goal would be to end in at least the top half. The only way to accomplish this, would be by implementing alot of ideas mentioned above. Whether using the framework of Team BanzAI, or only the knowledge we have accumulated through building it and participating at this year's competition.

## 6 Conclusion

The BanzAI system proved to have fair pathplanning and communication methods, but lacked use of coalitions (groups) and extensive use of strong central decision points, the center agents. Robocup 2003 at Padova proved that a system which is based on autonomous working agents can't achieve up to standard anymore. The V\*-algorithm has not become part of the framework, since low-complexity pathplanning is not necessary in most competition configurations. Robocup simulation league proves to be a useful testbed for AI-techniques, and team strategies tend to get closer to each other, an indication that the league is growing to its ultimate goal.

We could say that we failed at our task of creating a team of competent agents for the Robocup Rescue league, but we prefer to look at the positive sides. We learned alot by building the agents and competing and we saw other teams' agents and their extremely interesting ideas, which makes building a Team BanzAI II a possibility. Therefore we feel that our competition in the Robocup Rescue league was a success.

## Referenties

- [1] Stefan Leijnen Silvain van Weers, Christian Vossers and Andrew Koster. *BanzAI Team Description*.
- [2] Stefan Leijnen Silvain van Weers, Christian Vossers and Andrew Koster. *An advanced pathfinding algorithm*.
- [3] Takeshi Morimoto. *How to deveolop a RoboCupRescue Agent*.
- [4] Tomoichi Takahashi. *Robocup-Rescue Simulator Manual*.
- [5] Sebastien Paquet and Brahim Chaib-draa. *Pakrescue's Team Description*.
- [6] Cameron Skinner and Jonathan Teutenberg. *The Black Sheep Team Description*.
- [7] ResQ Freiburg. *ResQ Freiburg: Deliberative Limitation of Damage*.
- [8] Niusha Hakimipour Ehsan Mahmoudy, Siavash Mirarab and Ali Akhavan. *Eternity Team Description*.
- [9] M.L. Fassaert S.B.M. Post and A. Visser. *The communication reduction approach of the UvA Rescue C2003-Team*.