Final Report Context Project S&R

Group 2

June 20, 2014

Description of the developed functionalities

Robot handicaps

Part of the functionalities that we have developed is that robots in BW4T can now have certain abilities and inabilities, also referred to as handicaps, to broaden the possibilities of types of simulations.

The first handicap that we have developed is the *gripper handicap*, allowing us to set the number of *grippers* of a robot in a range from 0 - meaning the robot is not able to pick up any blocks - to 5, which could be described as a robot with a gripper and some sort of cart where it can store the blocks it picked up.

The second handicap is the handicap of *color blindness*, making the robot unable to distinct colors of different blocks. This handicap still allows to robot to see the blocks but it will make all blocks appear as dark gray blocks, simulating a robot with a black and white camera. Besides adding handicaps to a robot we can now also set different properties of the robot. In the scenario editor we can change the robot its size and speed, creating a robot that is unable to pass through doors because it is simply too large for the door, or making a really fast bot that can walk through the map at a much higher speed than other bots, possibly serving as a robot purely meant to scan the map and tell robots with multiple grippers where certain blocks are.

The last feature we have added to the robots is that robots can now carry a battery with them. The original bots can always continue walking around the map, picking up blocks and talking to their team members. However, when the user now chooses to set a battery for a robot (enabling it in the scenario editor) it can set its capacity and the bot will run out of it after moving around the map for a period of time. It can then recharge its battery by finding its way to a charging zone and recharge it. When the battery capacity is disabled the robot will always be able to keep moving.

Environment Store

The old map editor has been completely redesigned to what we now call the Environment Store. In this new Environment Store we are no longer limited to the standard mapping of rows and columns, bordered and separated by corridors with the start and drop zone at the bottom of the map. It offers the user the ability to create much larger maps and place the different types of zones across the map according to a grouping that is preferred for a simulation.

Besides the new functionality that allows us to set the side of the door in a room type of zone, we can now also add two new types of zones called blockades and charging zones. The blockades serve to create a map with parts where the robot should not be able to pass. This way the user can create a maze or zones that should be difficult or take a long time to reach. We have added charging zones which allow the bots that contain a battery to recharge when passing it. A charging zone is like an open space and does not contain any walls or doors. Both blockades and charging zones are the size of a basic room or corridor and therefore easily fit in the new mapping. In the new Environment Store we allow the user to randomize every aspect of the new map separately. Through the tools menu the user can randomize the rooms through a standard algorithm or randomize the blocks in rooms and sequence according to parameters the user can set himself. This way a complete map can be generated within seconds, even if the map is as large as a 1000 editable zones.

After the user has created a map, or while editing a map, it can at any time show a live preview of the real map by using the Preview map option in the file menu. This way the user can see how the changes in the map affect the actual environment as it would run in BW4T. When the user is satisfied with the map he has created, or simply wants to save the created map as a draft to continue with at a later stage, he can save and later open the map through a few simple clicks in the file menu.

Outlook

All in all, there are few things that the customer really wanted but we didn't implement because it was either too hard or simply impossible. Also, we were running short on time. The things that we didn't implement are the communication handicap (being able to communicate with another bot or not, or having a set chance that a certain message is going to be delivered) because this was too hard to do given the communication of GOAL-programmed agents, and the path finding and collisions of robots (robots can or cannot walk over each other in a path), because this was hard to do and we had time constraints. These are functionalities that can be added by new teams further developing the software. Also, the way we created handicaps allows for easy extension to more handicaps, although some old code has to be rewritten in order to pass the data to the data

object to be able to actually see the handicaps in action.

We think that the software we created can be easily extended in general to include more features, because of the massive effort to restructure the code and to create a framework for features. Aside from the functionalities which we didn't implement, ideas for a new version (which would be BW4T4) could be: more zones (such as repair zones for robots that are damaged by a collision), more hazards in the map that do different things (such as a supercharge zone, which overcharges robots that use a battery and damages them because of the overcharge, but robots without a battery would be unaffected), et cetera. A usability feature would be a button to randomize both the rooms and the block sequence at once, so that the map is still completable, because during the user tests there was a comment that randomizing both the rooms and the blocks/sequence was a little bit of a hassle. This would add more customization to the BW4T environment, and make it more user friendly, allowing it to be a better platform to test out GOAL agents that are going to be used in real life situations, the goal for developing the BW4T environment further.

We also made some errors with developing the software. For example, the option to load a map is not in the size dialog but in the map editor. This causes problems because the size of the map is already defined by then, and loading a map of a different size causes the obvious problems, especially when loading a map that is too big. It is also not logical to define the size of the map again before loading the map in the editor. Also, the bot store part of the system does not use the MVC design pattern anymore. It actually did after we refactored its code, but because group 3 was building their parts of the system around the old, non-MVC version of the bot store and we discovered this a day before the deadline, we decided that we would reset the bot store to the non-MVC compliant version. A true shame, but it was necessary because it was only shortly before the deadline and we had way too little time to fix and test everything. Better communication could have helped us here tremendously, but alas. A design issue for a new development group is therefore to recreate the bot store to an MVC compliant part of the system, and also rebuild the environment store to work with this new version.