Instructions for the Manual Client

Thomas Bolander & Mathias Nygaard Justesen

1 Running the client locally

The manual client can be used to control up to three different agents simultaneously on your own machine, i.e., you and up to two other students can collaborate to solve levels using the same keyboard. For levels with more than three agents, you have to control them on your separate computers via a webserver, see Section 2 below.

Run the manual client like any other client using the `-c` flag, e.g.:

```
java -jar server.jar -g 30
     -c 'java -jar manualclient.jar'
     -l levels/level-name.lvl <other options>
```

A window with information will be shown alongside the level. It briefly describes how to control the agents and the modes the agents are currently in. This is described in detail in Section 3. An example of such a window is shown in Figure 1. The window *must be in focus* to detect the keystrokes, which control the agents.

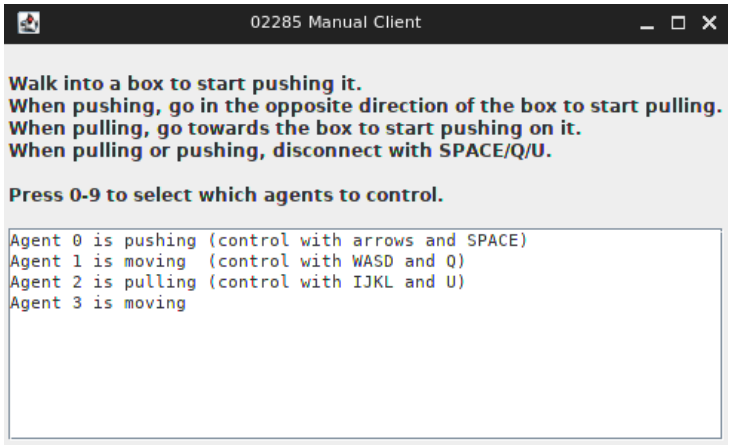

Figure 1: An example of the window shown with the manual client.

2 Connecting the client to a server

You can also collaborate to solve levels using multiple computers by connecting to a server via your browser. The server is located at `http://pc-02285.compute.dtu.dk/`

One of you have to create a *room* on the server, which is the one everyone else is connecting to. So pick a room name, e.g., your group name *`group-name`*, then start the client with

```
java -jar server.jar -g 30
     -c 'java -jar manualclient.jar group-name pc-02285.compute.dtu.dk'
     -l levels/level-name.lvl <other options>
```

Make sure to specify `-g 30` or no graphics to avoid lag when having multiple players connected.

Now, everyone can connect to the room via their browser. Decide on who controls which agent (e.g. you get to control agent *x*), then go to

`http://pc-02285.compute.dtu.dk/?room=`*group-name*`&agent=`*x*

In the browser there is a textual representation of the level as shown in Figure 2 (you have to initially press SPACE in the browser window to see this). This browser window *must be in focus*, so the server can detect your key strokes. The textual representation has more information than shown in the graphical representation of the normal Runner: It also shows whether each agent is in *pull*, *push* or *move* mode (see Section 3 for details on these modes). Still, we feel that it is easier to solve and keep track of the state of a level in the graphical representation, so even if your group is using the client in webserver mode, it is recommended that everyone is looking at the same Runner window on the screen of the computer that hosts the room.

```
++++++++++
+++c+++   +
+b+ +a+ + +
+      1+ +
+++0+++A+d+
++ C ++ +++
+++++++B+++
+3D    2+++
++++++++++
```

Figure 2: An example of the textual representation shown in the browser.

Each computer can control two agents, so if you prefer, you only need 2 computers to control 4 agents (and 4 computers to control 8 agents). You can control an extra agent *y* by adding `&agent2=`*y* to the URL.

3 Controlling the agents

An agent can be in one of three different modes: *move mode*, *push mode* or *pull mode*. This corresponds to the three different types of non-noop actions of the domain. Initially, all agents are in move mode. When controlling an agent via the webserver, the controls are as described for agent 0 below, and the controls of the extra agent (`agent2` in the URL) are as described for agent 2.

Move mode In move mode, the moves of agent 0 are controlled with `W`, `A`, `S`, and `D`, see Figure 3. Agent 1 is controlled with the arrow keys, and agent 2 with `I`, `J`, `K`, and `L`. In other

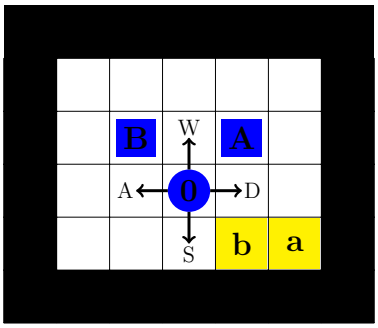

Figure 3: Controlling agent 0 in move mode with the `W`, `A`, `S`, and `D` keys.

words, in move mode the keystrokes translate into actions in the following way:

keystroke			becomes action
`W`	↑	`I`	$Move(N)$
`A`	←	`J`	$Move(W)$
`S`	↓	`K`	$Move(S)$
`D`	→	`L`	$Move(E)$

Push Mode When an agent is in move mode and attempts to move into a cell occupied by a box, it connects to the box and changes to push mode, see Figure 4. Note that the server doesn't visually indicate that the agent is connected to the box, nor that it is in push mode. However, the current mode of the agents is displayed in the information window for the manual client when run locally, and in the textual browser representation the agent symbol is boldface when run via the webserver.

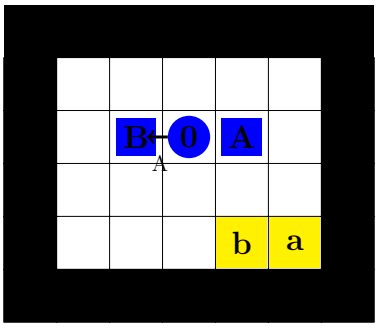

Figure 4: By pressing A, agent 0 connects to box B and goes into push mode.

In push mode, the keystrokes control the direction of the box connected to, see Figure 5. Hence, in push mode, the keystrokes translate into actions in the following way, where *curr-dir-*

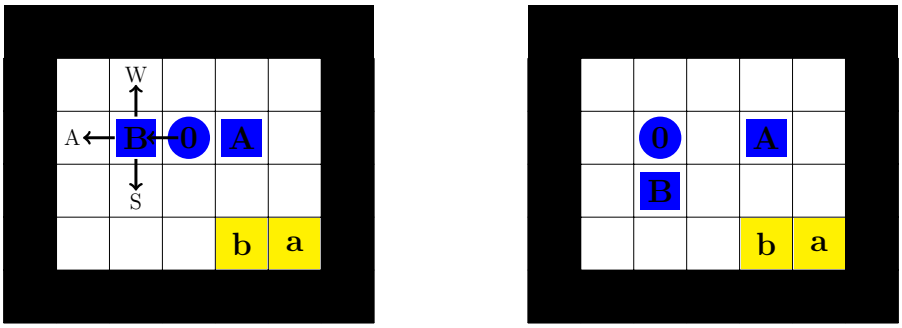

Figure 5: *Left*: In push mode, the keyboard controls the direction of movement of the box connected to. *Right*: After pressing `S`.

box is the current direction of the box connected to:

keystroke			becomes action
`W`	↑	`I`	$Push(curr\text{-}dir\text{-}box, N)$
`A`	←	`J`	$Push(curr\text{-}dir\text{-}box, W)$
`S`	↓	`K`	$Push(curr\text{-}dir\text{-}box, S)$
`D`	→	`L`	$Push(curr\text{-}dir\text{-}box, E)$

Note that one of these actions will never be applicable: the one where the agent attempts to move in the opposite direction of the box. Performing this action will bring the agent into pull mode. In the left state of Figure 5 this means that pressing `D` will enable the agent to pull box B.

The agent can disconnect from the box with `Q`, `SPACE`, or `U` for agent 0, 1, and 2, respectively.

Pull mode As mentioned above, when in push mode, the agent can change to pull mode by moving in the direction opposite of the box. In pull mode the keystrokes control the direction of the agent, and the connected box simply moves into the previous location of the agent. Again the server doesn't visually indicate that the agent is in pull mode, but it is shown in the information window for the manual client when run locally, and in the textual browser representation the agent symbol is in italics when in pull mode. Consider the situation in the right state of Figure 5, where the agent is in push mode and connected to B. Then pressing `W` will bring the agent into pull mode, and the keystrokes will now work as shown in Figure 6.

Hence, in pull mode, the keystrokes translate into actions in the following way, where *curr-dir-box* is the current direction of the box connected to:

keystroke			becomes action
`W`	↑	`I`	$Pull(W, curr\text{-}dir\text{-}box)$
`A`	←	`J`	$Pull(N, curr\text{-}dir\text{-}box)$
`S`	↓	`K`	$Pull(S, curr\text{-}dir\text{-}box)$
`D`	→	`L`	$Pull(E, curr\text{-}dir\text{-}box)$

As in the case of push mode, one of these actions will not be applicable: the one where the agent attempts to move in the direction of the box. Performing this action will bring the agent back into push mode. In the left state of Figure 6 this means that pressing `S` will enable the agent to push box B.

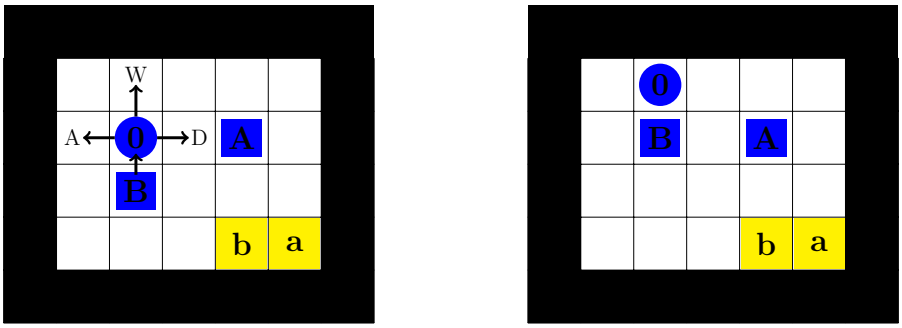

Figure 6: *Left*: In pull mode, the keyboard controls the direction of movement of the agent. *Right*: After pressing `W`.

The agent disconnects from the box with `Q`, `SPACE`, or `U` for agent 0, 1, and 2, respectively.

Changing which agents to control *The following only applies when running the client locally.* By default you control agent 0, 1, and 2 with `WASD` (and `Q`), the arrows (and `SPACE`), and `IJKL` (and `U`), respectively. If there are more agents in the level, you can change which agents to control with the numbers keys `0`–`9`. That is, if you push key n, you control agents n, $n+1$ and $n+2$ (modulo 10). Hence, pressing `4` will allow you to control agent 4, 5, and 6.