# The Blocks World Environment Manual

Koen V. Hindriks and W. Pasman

March 9, 2014

## 1 Introduction

The Blocks World environment that is distributed with Goal provides a graphical interface and simple environment for visualizing Blocks World problems in 3D. The Blocks World is a classic and famous toy domain in Artificial Intelligence. See, e.g., http://users.cecs.anu.edu.au/~jks/bw.html. Also see the Goal Programming Guide for a discussion of this domain and some example agents. The environment and various multi-agent system and agent files for the environment are included in the distribution of Goal.

## 2 The Blocks World Environment Interface

The environment interface file is named `blocksworld.jar`. Include a reference to this file in order to launch it. In a Goal MAS file include the reference in the **environment** section.

### 2.1 Initialization Parameters

The environment supports parameters for initializing the configuration of blocks and for disabling the graphical user interface that displays the environment. Include the parameters as `key = value` pairs in the **environment** section of a Goal MAS file by using the **init** command.

- **start** parameter: This parameter can be used to set the initial block configuration. The format that is used is based on a format introduced by Slaney ([1]; see also the url above). Either an explicit list can be used or a reference to a file that contains a specification of the configuration. A configuration is specified by a list of numbers that indicate on which other object a block should sit on. Zero 0 is used to refer to the table. The first number indicates where to put block 1, the second number indicates where to put block 2, etc.
  - A *list* should contain one number for each block separated by commas and the list should be surrounded by square brackets. For example, `[0,3,0]` means that block `b1` sits on the table, block `b2` sits on block `b3`, and block `b3` sits on the table. (Here we use the labels that are used by the environment to refer to a block.)
  - A *filename* should be inserted between double quotes (a string). The corresponding file should contain a plain list of numbers separated by white spaces without any punctuation. E.g., the file should be a simple text file containing `0 3 0` to generate the same configuration as with the list above. File names should either be absolute paths to the configuration file or relative paths to the folder where the environment interface jar file for the Blocks World is located.
  - If no start configuration is specified, the *default* is to initialize the environment with 8 blocks that sit on the table.
- **gui** parameter: This parameter can be set to true or false. If set to false, no GUI is opened so you can not see or manipulate the world. If no gui parameter is specified, the *default* is to display the GUI.

### 2.2 Actions and Percepts in the Blocks World

The Blocks World environment allows a gripper to move and stack various blocks on top of each other. An agent can control the gripper and thus move blocks. Blocks sit either on top of another block or on the table. Blocks are named `b<Nr>` where `<Nr>` is the block number. For example, `b1` is block number 1, `b2` is block number 2, etc. The table can be referred to by means of the constant `table`.

**Action** The gripper can perform one action in the environment called `move(X,Y)`. The first parameter `X` is the block that is moved and the second parameter `Y` is the object that the block is moved onto. A block can either be moved onto another block or onto the table. Only blocks that are clear can be moved; a block is clear if there is no other block that sits on top of it. A block can only be moved onto another block if that block is clear. A block that is clear can always be moved to the table (the table always has room to place a block). For example, `move(b1,b2)` would move block `b1` on top of block `b2` if both blocks are clear and `move(b9,table)` would move block `b9` to some free spot on the table if the block is clear.

**Percept** The environment provides a single percept of the form `on(X,Y)` for each block `X` that is present in the environment. The fact `on(X,Y)` means that block `X` sits on top of `Y`; `Y` can either be a block or the table. The Blocks World environment is *fully observable.* This is so because the set of facts that consists of a fact `on(X,Y)` for each block `X` present in the environment fully describes a configuration of blocks. These facts do not fix the exact location of a stack of blocks on the table, but this is considered irrelevant in the simple Blocks World.

## 3 User Interaction

You can interact with the Blocks World GUI by using the mouse and the keyboard.

A block can be moved by means of the **mouse**. To do so, select a block that is clear by clicking on it. A block gets a darker appearance when it is selected. After a block has been selected that block can be moved onto another block that is clear by clicking on that block. A block can be moved onto the table by not clicking on a block but somewhere else in the GUI.

Pressing the **space bar** moves all blocks onto the table. The following keys can be used to move the camera; pressing

- the X key moves the camera to the left,
- SHIFT-X moves the camera to the right,
- the Y key moves the camera downwards,
- SHIFT-Y moves the camera upwards,
- the F key moves the camera backwards,
- the N key moves it forwards,
- the L key increases the focal length (zoom in),
- the S key decreases the focal length (zoom out),
- the four cursor keys respectively pan the camera up, down, left and right.

# 4 Generating Initial States

The Blocks World is distributed with a program for automatically generating initial states. The program generates random initial states with a good random distribution [1]. The file that can be used is called `generatelearnfile.jar`. It can be run from the command line. To use the program, open a terminal and go to the directory containing the file `generatelearnfile.jar`. Then enter the following in the terminal:

```
java -jar generatelearnfile.jar <NrOfBlocks> <Seed> <NrOfStates>
```

Here, `NrOfBlocks` should be an integer that indicates the number of blocks that should be present in the Blocks World state or configuration, `Seed` should be a large integer number that you pick which is used to initialize the random number generator (e.g., 65237), and `NrOfStates` should be an integer number that indicates the number of states that should be generated. Note that the same seed number will generate exactly the same output.

Output will be printed in the terminal. E.g., issuing the command `java -jar generatelearnfile.jar 5 65123 20` will produce the output:

```
0,0,5,0,4
3,0,0,0,0
4,1,2,0,0
4,1,0,0,3
5,1,0,0,0
0,4,0,5,0
3,4,5,1,0
3,0,4,0,2
3,1,5,0,4
2,3,0,0,0
```

## Acknowledgement

This environment is based on code developed by Rick Wagner (wagner@pollux.usc.edu) at the University of Southern California. The source code may be used for educational purposes only.

## References

[1] Slaney, J., Thiébaux, S.: Blocks World revisited. Artificial Intelligence **125**, 119–153 (2001)