CS765 – Embodied Robotics Challenge

AIM: To get hands-on experience with design, implementation and analysis of an AI that employs cognitive systems and embodied robotics methods.

OVERVIEW

Working in a group of four, you will use **MALMO—**Microsoft's Minecraft-based platform for Artificial Intelligence research—to develop a simulated agent controller that demonstrates both cognitive systems *and* embodied robotics approaches that you have learned about in this class.

- Malmo homepage: https://github.com/Microsoft/malmo
- Malmo documentation: http://microsoft.github.io/malmo/0.14.0/Documentation/index.html

These instructions assume you have some (very basic) familiarity with Minecraft. If not, watch a youtube introductory video, ask around, and do some internet research to become acquainted with the game. Note that you do not need to know how to play minecraft to do this project, just make yourself familiar with the basics. After installing Malmo you could also launch a single-player game to briefly explore the game (without writing any AI)…but don't get sucked in and play for hours!

- Minecraft homepage: https://minecraft.net/

Note that what people do *in the game of Minecraft* (build shelters, fight zombies, etc.) does not need to be what you get your AI to do. You, as a team, get to decide how you think you can best take advantage of some of the techniques taught on this course to produce an impressive, interesting or entertaining AI. Some ideas to spark your imagination: you could create an AI that..

- responds to the needs of a human-player, e.g. to coordinate a group action (e.g. hunting)
- navigates a maze, while jumping over rivers of lava.
- uses simple sensors to distinguish between friendly and non-friendly entities
- shoot arrows at moving targets.

Note that you are expected to demonstrate BOTH cognitive systems AND situated / embodied / dynamical approaches. These do not need to be in the same agent—you can create different AI programs and environments for each system. Alternatively, you might want to compare two different approaches (one CogSys, the other SED) to solving the same task. Also note that you (as a team) are responsible for challenging yourself without over-challenging yourself.

The remainder of this document outlines the deliverables (what you will hand in and what will be assessed) before providing some instructions and tips for getting the most out of MALMO.

Additional documentation will also be made available that provides some tips and techniques that I have found useful while experimenting with Malmo to develop this challenge.

TIME-LINE

Monday 30 April (Week 8) Inform the course coordinator of your group membership and give a provisional (non-binding) project title. Do this by email to Matthew (m.egbert@cs.auckland.ac.nz). *Start thinking about a possible project topic and discuss ideas with your classmates!*

Monday 7 May (Week 9) Preliminary project proposal due.

Monday 28 May (Week 12) Project report due.

Monday 28 / Tuesday 29 May (Week 12) In-class presentation during week 11.

ASSESSMENT

Note that these deliverables will each receive a mark in a holistic form (not x out of y on a checklist of components) based on quality, largely as set out by the Law School's criteria for research-based work: see http://www.law.auckland.ac.nz/en/for/current-students/current-undergraduate-students/cs-course-planning/cs-regulations-policies-guidelines/marks-and-grades.html

1. **Proposal (5/20 points).** This document (1000 word maximum) should include:
    - an outline of the work you plan to accomplish. It is up to you to pick goals that are not-trivial, but also not too difficult to implement in the limited time you have available. In other words, be realistic about the scope possible for something you can work through to a demonstrable prototype in the time available – think 'narrow but interesting'.
    - a (brief) explanation of how your agent(s) will demonstrate both CogSys and SED approaches, including your *criteria for success* (i.e. what do you need to demonstrate to have succeeded in your project?)
    - references to related work that has inspired the challenges you are proposing
    - a brief explanation of what you expect to be interesting, impressive, or entertaining about your challenges & the solutions you develop (i.e. criteria for success)
    - work allocation—i.e., who in your group will carry out which parts of the work; (proposal writing, coding, testing, presentation, etc.)
2. **Presentation (5/20 points).** Using slides and video or live-demonstration, each group will present the results of their efforts to the class as a whole. You will be given **15 minutes** for your presentation, and approximately 5 minutes for questions from the rest of the class. There is no specific requirement on whether all members of the team speak – for instance, you may find that having two nominated speakers is most practical. Be careful to balance your time to include:
    - Challenge you gave yourself
    - Background & Design (what previous work did you extend?)
    - Evaluation (did your system succeed at the task you set for yourself)

3. **Report (10/20 points).** This document (2500 words maximum) should include:
    - Background. What research inspired your approach? Cite this research appropriately, and describe clearly any ways that your implementation differs from / extends previous work in this area.
    - Self-imposed challenges. This may borrow material from your proposal (assuming that your challenge has not changed). It should not repeat everything though. This section has a maximum length of 500 words (whereas the proposal has a 1000-word max).
    - Implementation & Evaluation. Explain what you built, and evaluate how well or poorly it worked. This is not a marketing exercise—you are not trying to sell what you have done, so marks are awarded here for honesty, not for pretending your system is more capable than it really is. Explain cases where it works, explain cases where it fails and if possible explain *why* it fails where it does.
    - You can upload a video demonstration to supplement your report if you wish (possibly identical to what you use for the in-class presentation), but this is optional.
4. Implementation. Your programs will not be directly assessed, but must be submitted digitally on Canvas at the same time as your report, so that we can further investigate your approach if we want to, and to check for plagiarism, etc.
    - I strongly recommend that you use Python for interacting with Malmo. Although they say they support a number of languages, I have not tested any except for Python, which appears to be the main supported Malmo language.
    - You are encouraged to leverage the power of existing specialised tools and libraries. Don't re-implement the wheel unnecessarily! Some example systems that you may be able to work with (not sure all of these are easily connected to Python).
        - CLIPS (C based expert system package)
        - JESS (Java based expert system package)
        - Pyke (inspired by Prolog; Python based)
        - Prolog (classic logic programming language - SWI Prolog has a Java API)
        - Lisp (classic list-based AI language; many versions and variants out there)
        - Jena and OWL (OWL [and RDF] is W3C standard for semantic specification – can edit with Protégé; Jena is Apache Java toolkit to reason on OWL)