ABOD3
A Graphical Visualisation and Real-Time Debugging Tool for BOD Agents

Andreas Theodorou
Department of Computer Science
University of Bath
Bath, BA2 7AY UK
Email: a.theodorou@bath.ac.uk

I. Introduction

Current software for AI development requires the use of programming languages to develop intelligent agents. This can be disadvantageous for AI designers, as their work needs to be debugged and treated as a generic piece of software code. Moreover, such approaches are designed for experts; often requiring a steep initial learning curve, as they are tailored for programmers. This can be also disadvantageous for implementing transparency to agents, an important ethical consideration [1], [2], as additional work is needed to expose and represent information to end users.

We are working towards the development of a new editor, ABOD3. It allows the graphical visualisation of Behaviour Oriented Design based plans [3], including its two major derivatives: Parallel-rooted, Ordered Slip-stack Hierarchical (POSH) and Instinct [4].

The new editor is designed to allow not only the development of reactive plans, but also to debug such plans in real time to reduce the time required to develop an agent. This allows the development and testing of plans from a same application.

II. Behaviour Oriented Design

Behaviour Oriented Design (BOD) [5], [6] takes inspiration both from the well-established programming paradigm object-oriented design (ODD) and Behaviour-Base AI (BBAI) [7], to provide a concrete architecture for developing complete, complex agents (CCAs), with multiple conflicting goals and mutual-exclusive means of achieving those goals. BBAI was first introduced by Brooks [7], where intelligence is decomposed into simple, robust modules, each expressing capabilities, actions such as movement, rather than mental entities such as knowledge and thought.

Bryson's BOD is a cognitive architecture which promotes behaviour decomposition, code modularity and reuse, making the development of intelligent agents easier. BOD describes the agent's behaviour into multiple modules forming a behaviour library. Each module can have a set of expressed behaviours called acts, actions, perceptions, learning, and memory. Behaviour modules also store their own memories, i.e. sensory experiences.

Action selection is forced by competition for resources. If no such competition exists, the behaviour modules are able to work in parallel enforcing the long-term goals of the agent

A. POSH

POSH planning is the action selection for reactive planning derivative of BOD. POSH combines faster response times similar to reactive approaches for BBAI with goal-directed plans. A POSH plan consists of the following plan elements:

1) Drive Collection (DC): It contains a set of Drives and is responsible for giving attention to the highest priority Drive. To allow the agent to shift and focus attention, only one Drive can be active in any given cycle.
2) Drive (D): Allows for the design and pursuit of a specific behaviour as it maintains its execution state. The trigger is a precondition, a primitive plan element called *Sense*, determining if the drive should be executed by using a sensory input.
3) Competence (C): Contains one or more Competence Elements (CE), each of which has a priority and a releaser. A CE may trigger the execution of another Competence, an Action Pattern, or a single Action.
4) Action Pattern (AP): Used to reduce the computational complexity of search within the plan space and to allow a coordinated fixed sequential execution of a set of Actions.

B. Instinct

The Instinct Planner is a reactive planner based on the POSH planner. It includes several enhancements taken from more recent papers extending POSH [8]. In an Instinct plan, an AP contains one or more Action Pattern Elements (APE), each of which has a priority, and links to a specific Action, Competence, or another AP.

III. The Plan Editor

The editor provides a customisable user interface (UI) aimed at supporting both the development and debugging of agents. Plan elements, their subtrees, and debugging-related information can be hidden, to allow different levels of abstraction and present only relevant information. The graphical representation of the plan can be generated automatically, and the user can override its default layout by moving elements to suit his needs and preferences. The simple UI and customisation allows the editor to be employed not only as a developer's tool, but also to present transparency related information to the end users, helping them to develop more accurate mental models of the agent.

Alpha testers have already used ABOD3 in experiments to determine the effects of transparency on the mental models formed by humans [9], [10]. Their experiments consisted of a non-humanoid robot, powered by the BOD-based Instinct reactive planner. They have demonstrated that subjects, if they also see an accompanying display of the robot's real-time decision making as provided by ABOD3, can show marked improvement in the accuracy of their mental model of a robot observed. They concluded that providing transparency information by using ABOD3 does help users to understand the behaviour of the robot, calibrating their expectations.

Plan elements flash as they are called by the planner and glow based on the number of recent invocations of that element. Plan elements without any recent invocations start dimming down, over a user-defined interval, until they return back to their initial state. This offers abstracted backtracking of the calls. Sense information and progress towards a goal are displayed. Finally, ABOD3 provides integration with videos of the agent in action, synchronised by the time signature within the recorded transparency feed.

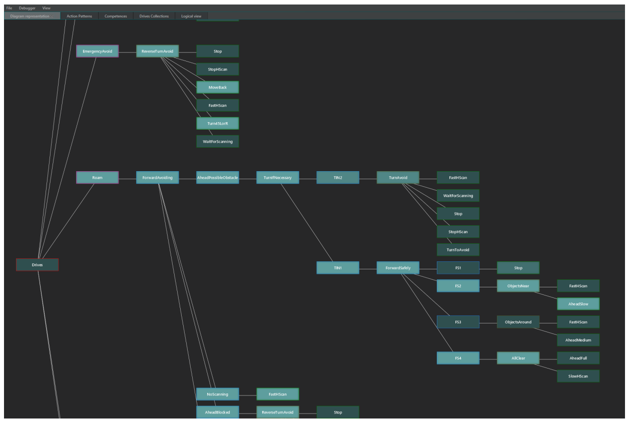

Fig. 1. The ABOD3 Graphical Transparency Tool displaying an Instinct plan in debugging mode. The highlighted elements are the ones recently called by the planner. The intensity of the glow indicates the number of recent calls.

ABODE3 provides an API that allows the editor to connect with planners, presenting debugging information in real time. For example, it can connect to the Instinct planner by using a built-in TCP/IP server, see Figure 2.

IV. Conclusion

We plan to continue developing this new editor, implementing debug functions such as "fast-forward" in pre-recorded log files and usage of breakpoints in real-time. A transparent agent, with an inspectable decision-making mechanism, could

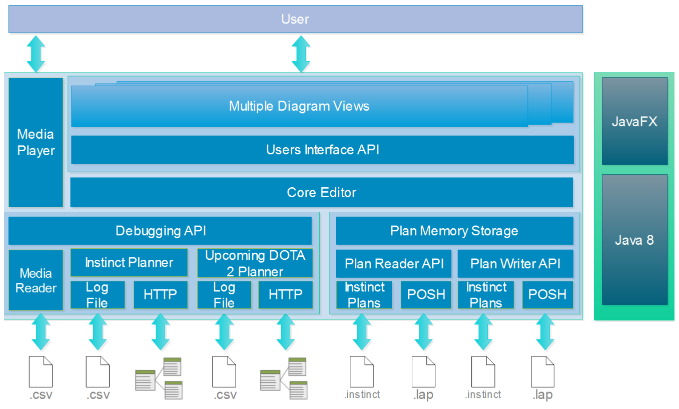

Fig. 2. System Architecture Diagram of ABOD3, showing its modular design. All of ABOD3 was written in Java, to ensure cross-platform compatibility. APIs allows the expansion of the software to support additional BOD planners for real-time debugging, BOD based plans, and User Interfaces. The editor aims, through personalisation, to support roboticists, games AI developers, and even end users.

also be debugged in a similar manner to the way in which traditional, non-intelligent software is commonly debugged. The developer would be able to see which actions the agent is selecting, why this is happening, and how it moves from one action to the other. This is similar to the way in which popular Integrated Development Environments (IDEs) provide options to follow different streams of code with debug points. Moreover, we will enhance its plan design capabilities by introducing new views, to view and edit specific types of plan-elements and through a public beta testing to gather feedback by both experienced and inexperienced AI developers.

References

[1] A. Theodorou, R. H. Wortham, and J. J. Bryson, "Designing and implementing transparency for real time inspection of autonomous robots," *Connection Science*, vol. 29, 2017.

[2] R. H. Wortham, A. Theodorou, and J. J. Bryson, "Robot Transparency , Trust and Utility," in *ASIB 2016: EPSRC Principles of Robotics*, 2016.

[3] J. Bryson, "The behavior-oriented design of modular agent intelligence," in *System*, 2002, vol. 2592, pp. 61–76.

[4] R. H. Wortham, S. E. Gaudl, and J. J. Bryson, "Instinct : A Biologically Inspired Reactive Planner for Embedded Environments," in *Proceedings of ICAPS 2016 PlanRob Workshop*, 2016.

[5] J. Bryson and L. A. Stein, "Intelligence by Design : Principles of Modularity and Coordination for Engineering Complex Adaptive Agents by," no. September 2001, 2001.

[6] J. J. Bryson, "Action selection and individuation in agent based modelling," in *Proceedings of Agent 2003: Challenges in Social Simulation*, D. L. Sallach and C. Macal, Eds. Argonne, IL: Argonne National Laboratory, 2003, pp. 317–330.

[7] R. A. Brooks, "Intelligence Without Representation," *Artificial Intelligence*, vol. 47, no. 1, pp. 139–159, 1991.

[8] S. Gaudl and J. J. Bryson, "The Extended Ramp Goal Module: Low-Cost Behaviour Arbitration for Real-Time Controllers based on Biological Models of Dopamine Cells," *Computational Intelligence in Games 2014*, 2014. [Online]. Available: http://opus.bath.ac.uk/40056/

[9] R. H. Wortham, A. Theodorou, and J. J. Bryson, "What Does the Robot Think? Transparency as a Fundamental Design Requirement for Intelligent System," in *IJCAI-2016 Ethics for Artificial Intelligence Workshop*, 2016.

[10] R. Wortham, A. Theodorou, and J. J. Bryson, "Improving robot transparency:real-time visualisation of robot ai substantially improves understanding in naive observers," in *IEEE RO-MAN 2017*, 2017.