NUbugger: A Visual Real-Time Robot Debugging System

Brendan Annable, David Budden and Alexandre Mendes

School of Electrical Engineering and Computer Science
Faculty of Engineering and Built Environment
The University of Newcastle, Callaghan, NSW, 2308, Australia.
{brendan.annable,david.budden}@uon.edu.au
alexandre.mendes@newcastle.edu.au

Abstract. As modern autonomous robots have improved in their ability to demonstrate human-like motor skills and reasoning, the size and complexity of software systems have increased proportionally, with developers actively working to leverage the full processing performance of next-generation computational hardware. This software complexity corresponds with increased difficulty in debugging low-level coding issues, with the traditional methodology of inferring such issues from emergent high-level behaviour rapidly approaching intractability. This paper details the development and functionality of NUbugger: a visual, real-time and open source robot debugging utility that provides the user with comprehensive information regarding low-level functionality. This represents a paradigm shift from corrective to preventative debugging, and concrete examples of the application of NUbugger to the identification of fundamental implementation errors are described. The system implementation facilitates simple and rapid extension or modification, making it a useful utility for debugging any similar complex robotic framework.

Keywords: debugging, robotics, open source, visualisation

1 Introduction

The problem of developing a team of humanoid robots capable of defeating the FIFA World Cup champion team, coined "The Millennium Challenge" [7], has been a milestone that has driven research in the fields of artificial intelligence, robotics and computer vision for over a decade. Corresponding with the continual improvement in a robot's ability to demonstrate human-like motor skills and reasoning is an exponential blowout in software size and complexity, facilitated by the evolution of robot platforms and subsequent advances in processor performance (from the 384 MHz RISC-based processors of the Sony AIBO ERS-210 (2002) to the 1.6 GHz Intel Atom processors of the Robotis DARwIn-OP [5] platform (2012)); a trend often inferred from Moore's Law [11].

As with any system of software or hardware, exponential increases in system size and complexity necessitate the introduction of hierarchial layers of abstraction, allowing low-level functionality to be handled transparently by higher-level