Robotics and Computer-Integrated Manufacturing 27 (2011) 292–305

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Full length Article

A generic controller architecture for intelligent robotic systems

Michael Short [a,*], Kevin Burn [b]

[a] Electronics & Control Group, Teesside University, Middlesbrough, TS1 3BA, UK

[b] Department of Computing, Engineering & Technology, University of Sunderland, Sunderland, SR1 3SD, UK

ARTICLE INFO

Article history:
Received 4 June 2007
Received in revised form
20 July 2010
Accepted 27 July 2010

Keywords:
Robotics
Controller architectures
Intelligent control
Force control

ABSTRACT

This paper describes a research program to develop a novel reference architecture and design philosophy for an advanced robot controller for a new generation of robots. It is suggested that a new approach to robot controller design is required in order to bring the present generation of industrial robots in line with current and foreseeable technological developments. The paper also describes a prototype controller that has been developed using this design philosophy, and its deployment on an experimental robot. Practical results are presented from a series of investigations undertaken to illustrate the performance of the controller.

1. Introduction

Industrial robots are currently employed in a large number of applications and are available with a wide range of configurations, drive systems, physical sizes and payloads. However, the numbers in service throughout the world are much less than predicted over 20 years ago [1]. This is despite major technological advances in related areas of computing and electronics, and the availability of fast, reliable and low-cost microprocessors and memory. This situation is mainly a result of historical and economic circumstances, rather than technical considerations. Industrial robots have traditionally performed a narrow but well-defined range of tasks to a specified degree of accuracy, and whilst new robot arm designs are specified for many years of continuous operation, the technological development of their controllers has been slow in comparison with other computer-based systems. As a consequence, most present-day controllers do not fully utilize current levels of technology, severely limiting the range and diversity of future, more advanced applications.

In contrast, much academic research has been undertaken aimed at improving the performance of robots using advanced control methods. They have included model-based techniques for adaptive control [2,3], force and hybrid force/position control schemes [4] and intelligent control methods, especially using artificial neural networks (ANNs) and fuzzy logic control [5,6]. Whilst varying degrees of success have been demonstrated, the application of many advanced methods has often been severely restricted in commercial systems by limitations of their controllers rather than their manipulator arms [7]. Nevertheless, modern commercial robotic systems are still highly complex. They integrate multiple sensors and effectors, have many interacting degrees-of-freedom (DOF) and require operator interfaces, programming tools and real-time capabilities [8].

This paper describes a research program to develop a novel robot controller for a new generation of robots. It is structured as follows. Previous work in the area is briefly summarized in Section 2. The concept of an open architecture, user-friendly and 'intelligent' sensor-based robot is then introduced in Section 3, and a realistic description of how this concept can be realized is provided in the form of a generic reference controller architecture. This is aimed at meeting the requirements of a wide range of robot users, including system developers, end users and research scientists. In addition to utilizing many established features of the current generation of devices in operation, it also enables the implementation of more advanced control techniques, including intelligent methods.

The design philosophy takes into account the need for continual upgrading as new hardware and software components become available, thereby preventing the controller from becoming obsolete in the face of new technological developments. Ultimately the goal is to develop a platform that provides robots with the versatility to perform a much wider range of tasks in a more diverse range of applications. In order to evaluate the effectiveness of this design philosophy, Section 4 describes a prototype controller that has been developed and employed

* Corresponding author. Tel.: +44 7973775274.
E-mail addresses: m.short@tees.ac.uk, michael.short@uk2.net (M. Short), kevin.burn@sunderland.ac.uk (K. Burn).

0736-5845/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.rcim.2010.07.013

on an experimental manipulator arm. In Section 5 we present experimental results from a series of tests that illustrate the performance of the controller in a variety of tasks. Finally, the effectiveness of the overall design is summarized in Section 6, and areas of further work are described.

2. Background work

2.1. Technology assessment

In the 1970s, technology limitations meant that robot controllers were usually based around bulky microcomputers, such as the DEC LSI 11/02. As technology has advanced, robot manufacturers have begun to utilize more compact, faster microprocessors, including specialized architectures such as transputers and systolic arrays [9]. It is therefore not surprising for a modern commercial robotic controller to make use of the PC-based technology. The popularity of PC architectures, and operating systems such as Windows, has arisen because of its perceived 'openness'. The computer workstation community first coined the phrase 'open architecture' in the early 1980s, the meaning of which is captured by the following definition [10]:

> *"an architecture whose specifications are public. This includes officially approved standards as well as privately designed architectures whose specifications are made public by the designers"*

In general, the advantage of open systems is that research engineers and third-party companies can develop and supply hardware and software, increasing the capabilities of the product; this increases the customer base, leading to more versatile low cost devices becoming available. With this definition in mind, robot controllers can be broadly classified into three types [11,12]:

- Proprietary: the controller structure is effectively closed. Integration of external or new hardware (including sensors) is either very difficult or impossible.
- Hybrid: the majority of the system is closed (e.g. in terms of control laws) but some aspects of the system remain open. It is possible to add – albeit in a limited fashion – new devices such as sensors and integrate them into a robot control program.
- Open: the controller design is completely available to be changed or modified by a user. The hardware and software structure can be changed such that all elements (servo laws, sensors, user interfaces, etc.) can be modified without difficulty.

Despite the availability of PC-based architectures and 'open' operating systems, the vast majority of commercial robot controllers – even when based around these open technologies – are still of the proprietary or hybrid type. If a user has an application that is well defined and unlikely to change (as in many factory applications of robots), then this type of controller is desirable as an off-the-shelf product. Many controllers, although now more technologically advanced when compared to earlier systems, are supplied for a particular manipulator (or product range of manipulators) and remain largely 'closed' in terms of access to servo control parameters and control laws. This can act to severely limit the range and diversity of future, more advanced applications.

2.2. Existing reference architectures

The majority of research undertaken to address the problems outlined in the previous section has been led by the space industry and has been concerned with the robot-specific interpretation of government standards regarding general 'open architecture controllers'. This has resulted in a multitude of projects and by way of illustration, two important examples are briefly discussed here: the National Aeronautics and Space Administration/National Bureau of Standards (NASA/NBS) Standard Reference Model for Telerobot Control System Architecture (NASREM) and the European Space Agency's (ESA) Control Design Methodology (CDM) [13,14].

The NASREM architecture is a conceptual model that decomposes into three hierarchical levels, each of which performs a fundamental mathematical transform. Each level is split into three sections – task decomposition, world modelling and sensory processing – and each section is a finite state machine that accepts input, performs a transformation based on input and state and produces output. The input consists of higher-level commands, sensory data from the same level and status data from lower levels.

The ESA-CDM was heavily influenced by NASREM. It defines three steps in the design process for automation and robotics applications. The first is called activity analysis and aims to define precisely the tasks the robot has to accomplish. Functional analysis is the second step and establishes the control functions required and human–machine interaction (HMI) that is needed to accomplish the tasks defined in step one. The third step is the architectural design phase in which the controller software and hardware are considered in detail.

The CDM utilises a functional reference model (FRM) for the activities of robot control systems and this is shown in Fig. 1. The top hierarchy, the MISSION layer, attempts to describe the activities that the robot is responsible for in very abstract terms, for example, SERVICE a satellite and REPAIR a platform. The TASK layer decomposes the high level activities into tasks, which are defined as the highest level of activity that can be performed on a single subject/object (OPEN a door, WELD a seam, etc.). Finally, the lowest level sub-divides each individual task into different

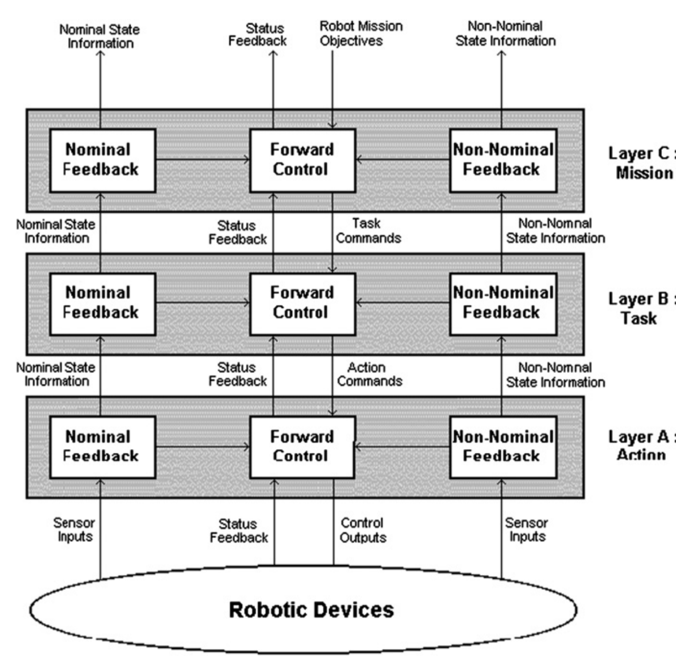

Fig. 1. The ESA-CDM architecture.