International Journal of Industrial Ergonomics 53 (2016) 274–283

Contents lists available at ScienceDirect

International Journal of Industrial Ergonomics

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

Development and evaluation of a virtual training environment for on-line robot programming

Dimitris Nathanael*, Stergios Mosialos, George-C. Vosniakos

School of Mechanical Engineering, National Technical University of Athens, Zografou, Greece

ARTICLE INFO

Article history:
Received 16 July 2015
Received in revised form
3 December 2015
Accepted 22 February 2016
Available online 7 March 2016

ABSTRACT

The paper reports on the development and evaluation of a virtual reality system to support training in on-line programming of industrial robots. The system was evaluated by running training experiments with three groups of engineering students in the real, virtual and virtual augmented robot conditions. Results suggest that, the group with prior training in the virtual reality system augmented with cognitive/perceptual aids clearly outperformed the group that executed the tasks in the real robot only. The group trained in the non-augmented virtual reality system did not demonstrate the same results. It is concluded that the cognitive/perceptual aids embedded in the augmented virtual reality system had a positive effect on all task performance metrics and on the consistency of results across participants on the real robot. Virtual training environments need not be designed as close as possible to the real ones. Specifically designed augmented cognitive/perceptual aids may foster skill development that can be transferred to the real task. The suggested training environment is simple and cost effective in training of novices in an entry level task.

1. Introduction

Virtual reality systems provide a valuable tool for training in various industries, where either the cost or the possible negative consequences of exposing trainees to the real task environment are considerable.

Spatial skill and procedural learning transfer from a virtual to the real environment has been reported to be positive in several cases examined (Regian, 1997; Waller et al., 1998; Aurich et al., 2009). Nevertheless, even when a clear transfer of training occurs, it is probable that the overall effect of training will mask a mixture of more specific effects, some of which facilitating correct real world performance (positive transfer) whereas some other hindering it (negative transfer). There is widespread belief that the main challenges for Virtual Reality training effectiveness and applicability have to do with the necessary physical fidelity to mimic the resolution of the physical world (Gupta et al., 2008; Slater and Wilbur, 1997). This view is however probably limited. In effect, some of the most successful Virtual reality training systems such as the MIST VR surgical simulator (Gallagher et al., 1999) have been highly successful even though they are judged as of very low fidelity by today's standards. In addition, simulators not merely acting as real world replacements have had considerable success in transfer of training by making creative use of the possibilities offered by virtual environments e.g. by flying around a building instead of walking or making use of transparent walls (Lathan et al., 2002). Bardy et al. (2012) propose that the value of a training system should be judged (i) by its ability to provide relevant experience, (ii) by the provision of facilitation and guidance to the acquisition of the designated skill and (iii) by the transfer from VR training to performance in the real world. Therefore, relevance, facilitation and transferability are the key constructs and the crucial evaluation criteria for a training system. In order to become effective then, VR training should be oriented towards the establishment of what it is that is being transferred from the virtual to the real environment (Rose et al., 2000).

Industrial robots have been chiefly employed in the last three decades for material handling, e.g. tending machine tools, but also for manufacturing processing, e.g. welding, deburring etc., as well as for assembly. Their programming and re-programming is time consuming or even tricky in the case of complex manufacturing scenarios but it is the most important task to be performed throughout their life.

Traditionally, programming systems for industrial robots can be

---

* Corresponding author.
E-mail address: dnathan@central.ntua.gr (D. Nathanael).

http://dx.doi.org/10.1016/j.ergon.2016.02.004
0169-8141/© 2016 Elsevier B.V. All rights reserved.

divided into three categories: (a) guiding systems, where the robot is manually moved to each desired position and the joint coordinates are recorded, (b) robot–level programming systems, typically employing a relatively low-level programming language provided with the robot and (c) task–level programming systems, functioning at a higher level, whereby the goals to be achieved rather than the moves as such are specified. In the case of more sophisticated non-industrial robots with embedded intelligence, programming may involve the use of graphical or text–based programming languages and automatic programming, including learning systems, programming by demonstration and instructive systems (Biggs and MacDonald, 2003).

In industrial practice, guiding systems are employed most often (usually termed 'on-line', 'teaching' or 'lead-through' systems), but note that the 'taught' positions are used together with controller-specific language commands that are manually entered to generate a program. The complete path is visualised by executing the program on the robot as such or on a simulator. Calculations and complex logic are generally not straightforward to integrate into such programming systems.

Modelling the robotic cell on a simulator and subsequently programming the robot on it (usually termed 'off-line' programming) offers advantages, such as continuously available visualisation, parametric definition of the path (Zlajpah, 2008) and, most notably, no need to keep the robot from executing its normal tasks while programming it. User interaction is generally required (Jara et al., 2011) but optimisation algorithms and tools may be reverted to in order to automate some of the interactive tasks (Chen et al., 2003).

Simulators are based on CAD modellers (Pires et al., 2004) and may exploit relevant functionality, notably constraint-based modelling (Vosniakos and Chronopoulos, 2009). However, even if geometrically accurate models of all equipment elements are available, simulation concerns purely kinematics and does not encompass dynamics and control models, which would have allowed a behaviour close to the real robots. Thus, the robot path derived by off-line programming may need to be corrected on the real robot according to calibration and other procedures that may be time consuming, too (Angelidis and Vosniakos, 2014).

Simulators based on Virtual and Augmented Reality (VR-AR) were initially simplistic (Burdea, 1999), but more recent developments in VR/AR are increasingly making an impact (Pan et al., 2012). Purpose-built VR environments in-lieu of previous generation CAD-based simulators have been reported (Gogouvitis and Vosniakos, 2014). Multimodal interfaces such as CAVE, Head Mounted Display (HMD), 3D haptic devices and force/acceleration sensors have already been employed in off-line programming of complex manufacturing scenarios (Mogan et al., 2008) (Haton and Mogan, 2008). Path planning decisions are supported in VR/AR by indicating collision-free volumes (Chong et al., 2009), by presenting alternative collision free paths (Hein and Worn, 2009), by fitting trajectory curves to just a few points through learning algorithms (Fang et al., 2012) etc. In addition, interesting AR interfaces are emerging, concerning, for instance, effective definition of robot operations at the task level using real workpiece data and process limits (Reinhart et al., 2008), specification of end-effector orientation observing dynamic constraints of the robot (Fang et al., 2012), facilitation of task recognition through virtual fixtures, both visual and tactile, in a programming by demonstration paradigm (Aleotti et al., 2004).

The notable advantage of VR, but especially of AR and mixed reality (MR), approaches is that they essentially allow intermingling off-line and on-line programming. In particular, they enable the user to manipulate a digital model of the robot, at the same time enhancing cognition either through added information via extra models or calculations (AR) or through presenting parts of the real world (MR). There are obvious cost benefits to such approaches compared to experimenting with the real objects but the most significant added benefit is the enhanced information content. However, most of these benefits have been reaped in a quest to replacing the robot operator by novel programming systems rather than to enhancing the pertinent skills of the operator by novel training systems.

Training systems for robotics programming are in essence concerned with spatial skills including motion synthesis and analysis (Verner et al., 2012). According to cognitive scientists, IT can facilitate effective training spatial skills in different contexts (Péruch et al., 2000). In order to design effective, yet generic enough training systems for robot programming, a mapping between skills and tasks is necessary. Training is normally focused on a single task, a family of tasks or, better still, on a taxonomy of tasks, such as the taxonomy of assembly tasks defined by Huckaby and Christensen (2012).

VR/AR based training systems pertaining to technical equipment exist in abundance. However, there are very few reports of such systems in the context of robot programming. As an example, in the context of space robotics operator training, AR was used to reduce positioning errors and time to completion of manoeuvring tasks with inherent poor visibility conditions. Specific overlay symbols were designed to help in alignment within insertion tolerances, to prompt appropriate control command motions and to allow separation of necessary translation and rotation control inputs (Maida et al., 2007). In the neighbouring field of assembly skills training, Adaptive Visual Aids have been proposed consisting of a tracking-dependent pointer object and a tracking-independent content object instead of traditional AR overlays (i.e. detailed 3D models or animations), which suffer from tracking inaccuracies (Webel et al., 2013).

In this work a VR system to support training in on-line programming of industrial robots is presented. Section 2 presents the reasoning behind planning and implementation of training using this environment. The development of the (VE) is outlined in Section 3. Use and evaluation of the training environment is focused on in Section 4 by presenting and analysing the results of an experiment designed to this end. Section 5 summarises the conclusions of this work.

2. Specific aims of training

Lead-through programming involves manipulation of the robot by means of a teach pendant to perform movements in three complementary coordinate systems; the Joint, World and Tool system and associated control modes. In the Joint system control mode, individual specified Joints of the robot, typically rotary, are moved about their pivot axes; in the World and Tool control modes the robot's end-effector is moved with reference to a Cartesian system which is either fixed in 3D space (World mode) or fixed on the moving end-effector (Tool mode). The three coordinate systems are complementary in the sense that the operator must combine all three modes to effectuate the desired robot movement across the robot's work volume envelope. This requires from the operator to acquire the ability to anticipate control mode effects in all three coordinate systems and the consequent ability to frequently shift focus among them.

A generic cognitive task analysis of programming a robot through the teach pendant has been suggested by Gray et al. (1992). This analysis involves four basic planning tasks, (i) select path from A to B, (ii) select programming mode, (iii) plan move and (iv) select controls. The first three tasks are essentially performed in a cyclical manner, by anticipating different alternatives through mental