Hammer: An Android Based Application for End-User Industrial Robot Programming

Carlos Mateo, Alberto Brunete, Ernesto Gambao, Miguel Hernando
Centre for Robotics and Automation (CAR UPM-CSIC)
Universidad Politecnica Madrid
Madrid, Spain
Email: ernesto.gambao@upm.es

***Abstract*—This paper presents a novel tablet based end-user interface for industrial robot programming (called Hammer). This application makes easier to program tasks for industrial robots like polishing, milling or grinding. It is based on the Scratch programming language, but specifically design and created for Android OS. It is a visual programming concept that allows non-skilled programmer operators to create programs. The application also allows to monitor the tasks while it is being executed by overlapping real time information through augmented reality. The application includes a teach pendant screen that can be customized according to the operator needs at every moment.**

Keywords—industrial robot; end-user programming; android OS;

I. INTRODUCTION

Currently only about 3% of the overall industrial robots employed in industry are used for machining. This is quite in opposition to the market potential and benefits of the industrial robot machining applications. It has been widely recognized that inherent 5+ axis machining capability combined with flexibility, large work envelope, multiple station capability and appropriate HMI is a flexible solution that allows end-user to expand the range of machining applications at a price much competitive to that of employing a traditional CNC machine [1]. With the recent dynamic cost reduction and performance optimization of modern industrial robots, the price of a comparable robotic solution is typically 1/5-1/3 of the cost of a CNC machine. Integration of two or more robots into flexible multi-stationed and multi-tolled robotic machining cells may result in significantly lower cost investments in comparison to employing large CNC machines. Based on several studies, the two underlying technical limitations for a widespread adoption of robotic machining are insufficient robustness of robotic structures (insufficient precision and stiffness) and lack of efficient programming tools that transfer CAD models into robot motion.

The work presented in this paper has been developed in the Hephestos project framework [12]. Hephestos' main objective is to develop novel technologies for the robotic hard material removal that will provide standard industrial robots with advanced techniques in production planning, programming and real-time control, and make available a promising and practical use of industrial robots for machining applications that is not possible at present.

One of its objective is to improve and make robot planning and programming efficient and promising for batch production of all scales. And to optimally combine standard planning tools, robot and process models with sensory feedback and human knowledge and experience in order to reduce the planning and programming costs by means of intuitive novel programming methods (e.g. manual or sensor guided).

In this sense, it is necessary a new tool to integrate novel and efficient intuitive on-line programming methods (e.g. programming by demonstration, manual-guidance, force-based shapes tracking, etc.), as well as sensory feedback about actual machining state in order to support re-planning and reprogramming. APIs must provide a human-interface for technological instructions and rules to obtain a more efficient and feasible programming. Also, an augmented reality programming environment would facilitate testing, evaluation and optimization of manufacturing programs and system real-time machining performance based on real robotic system and virtual interaction metal-removal task models (virtual force sensor).

The application presented in this paper is a simple (but powerful), intuitive and efficient robot programming environment that can be used by a non-skilled programmer and that can be easily integrated with the robot control and sensor systems. It provides a safe and flexible human-machine interface for dynamic cooperation to support on-line programming, efficient work-piece alignment, as well as human expert knowledge inclusion for an efficient programming.

II. RELATED WORK

A big effort has been put in the development of advanced human-robot interface systems for industrial robot environments. New technologies and techniques such as voice [2] or gesture recognition [3] [4] have contributed to the development of new HMI systems.

Taking into account real industrial programming environment needs, the use of advanced graphical user interfaces (GUI) is considered the most appropriate way for human-robot interaction in the programming phase. Modern portable devices such as PADs or tablets that include very precise and high resolution touch screens are available in the market at a very reduced cost. These devices are very appropriate in developing simple drag-and-drop introduction of commands, simplifying the programming procedures and reducing the

efforts in these tasks. There are also rugged tablets designed for dirty environments. Additionally, these devices can be easily connected to other computer systems, allowing a simple integration with the robot control and sensor systems. GUIs offer simplicity and a high degree of flexibility allowing to show the operator only what he really needs and reducing efforts and possible errors. In the other side, voice and gesture recognition cannot be simplified in a similar way and do not offer, in general, good performance in real industrial noisy and often populated environments.

The application presented here is based on the concepts of Google App Inventor and Scratch [15]. They are educational programming languages and multimedia authoring tools that can be used by anyone familiar with computer programming to create software applications. They use a graphical interface that allows users to drag-and-drop visual objects to create an application.

Another application that is based on Scratch is Catroid (now Pocket Code) [5], a free and open source visual programming system that allows casual and first-time users starting from age eight to develop their own animations and games solely using their Android phones or tablets. Catroid also allows wireless control of external hardware such as Lego Mindstorms robots via Bluetooth, Bluetooth Arduino boards, as well as Parrot's popular AR. Drone quad-copters via Wi-Fi. This application is not for industrial robots.

Other projects that uses Android applications to communicate with robots are [6] and [7]. Delden et al. [6] introduce an android platform that communicates with robots over a Bluetooth connection, so a user can control several robots at the same time, so the user does not have to obtain access to each robot teach pendant or terminal. Yepes et al. [7] present an Android OS based application that communicates with an industrial robot Kuka KR-6 through USB to Serial connection, to control it with the on-board accelerometers, and gyroscopes of a tablet or smart-phone, intended to be used in telemedicine procedures. Arduino Uno micro-controller board, RS232 Shifter SMD and mobile device were used to develop this work. But these two systems are not programming tools.

About the suitability of using Android devices to control robots, Neira et al. [8] presents a flexible solution that can be incorporated in most of the Android devices in the market, implemented and tested in a manufacturing scenario by creating adaptive interfaces for different types of user based on the user roles, tasks, the state of the system and the context. Nicolae et al. [9] studies the utilization of PDAs and mobile phones as human machine interface (HMI) in controlling various systems, and analyses the limitations of balancing processing load between process controller and Android device's resources, concluding that it is possible.

Regarding tablets as teach pendants, Jan et al. [10] propose a smart phone based teaching pendant that provides a user friendly interactive control input method to the robot's operator. The operator can not only give commands to the end effector, but during the continuous mode operation, the operator can pause, repeat and restart the subtasks of whole operation remotely. The two way network socket communication running on threads also gives a real time feedback data for detailed monitoring.

Augmented reality tablet applications are considered in [11]. It presents an idea of augmented reality based teaching pendant on smart phone and stating that incorporation of augmented reality into smart-phone based teaching pendant will help user to program industrial robot more intuitively.

It is possible to conclude that although the use of Android tablets as teach pendants has yet to take off, several projects mentioned in the state of the art have proven its feasibility.

III. Application description

Programming a robot is nowadays a hard task because a knowledge of each robot programming language and its set of instructions is needed. The primary goal of Hammer is to reduce all this work to find a simple intuitive block's language which let any person who has basic programming knowledge to make a program which will be send to the robot without needing to know its specific language. In this way it is possible to program robots without learning every robot specific language.

The Hammer App is based on end-user programming techniques, and more specifically on visual programming. It has been programmed for Android devices. It is able to run on any Android device, but it is especially designed for tablets running android from version 4.0 (Ice-cream sandwich).

As previously mentioned, the application has been based on the concepts of Google App Inventor and Scratch, but applied to robotic control and HRI, and specifically adapted to robot machining tasks. The application is designed for on-line programming/reprogramming and to be intuitive, easy-to-use and simple; easy use of learn by demonstration methods; easy connection with the robot control and sensors systems; and safety system integration

Another functionality provided by Hammer is direct and inverse kinematic control through a virtual interface similar to a teach pendant. It allows visualization of a simulated robot world and generation of paths and points to use in the programming interface previously mentioned.

Finally it should be mentioned that algorithms of augmented reality have been implemented to allow showing robot execution data overlapped in the images provided by the device's camera, e.g. visual representation of the force and torque vectors of the end effector.

The app has three main parts that will be described in more detail in the following sections: customized teach pendant, robot programming IDE and augmented reality based monitoring system.

IV. Robot Programming IDE

A. Description

The main feature of Hammer is to allow robot programming in an intuitive way, because Hammer internally would encode these instructions and would translate them into specific robot language.

Once the program is created, it can be executed in a simulation in the same device, or sent to other simulation software (e.g. in the Hephestos project it is used EasyRob

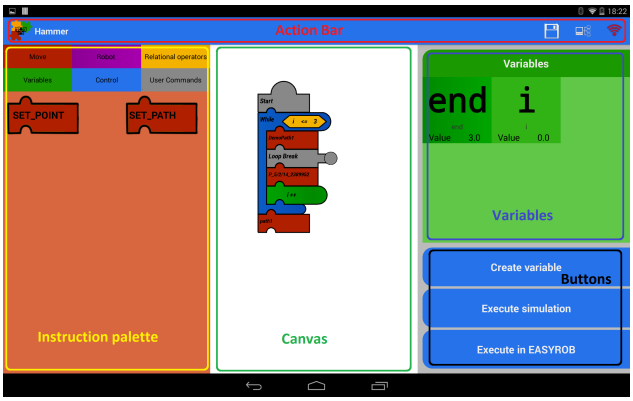

Fig. 1. Scratch View

Fig. 2. Loading Environment View

software [13]) which would execute the program with its own models.

The programming interface is structured in 5 main parts as shown in Fig. 1:

- Instructions palette: this part contains the visual blocks (i.e. instructions) grouped in 6 sections: movement, robot, relational operators, variables, control and user commands. These sections and their instructions will be described in the programming language section.
- Action bar: allows to save the created program or configure the port's connection to a computer running third-party simulation software.
- Canvas: this is where the program is developed, and where the instruction blocks are placed by drag and drop from the palette. Once in the canvas, blocks can be moved over each other to get linked.
- Variables: contains the created variables that can be used in the program. It shows the variable's name and its value. Variables can be deleted by a long tap.
- Buttons: executes the instructions of the program. There are two different ways: External execution, to send the program to the robot or to an external simulator (EasyRob) software to execute it, and Simulate, to show a simulation of the program generated in the device.

B. Loading Environment

A 3D environment created using XML files can be loaded into the Hammer application. It is possible to load CAD models with their own paths and points associated. Associations have to be declared in the XML file.

Fig. 2 shows the loading environment interface. It is structured in the following parts:

- XML list: shows the available XML files to load in the 3D canvas.
- 3D Canvas: shows the XML environment loaded. If an object is selected, the paths and points associated to it in the path list are also displayed.
- Path list: displays the paths and points of the selected object. By sliding this section to the right and left, different lists can be selected. If one of the list's items is selected, it will be shown in the 3D canvas.

C. Programming language

The programming language is based on visual programming with blocks that are divided in 6 groups. To create the program, blocks have to be dragged and dropped in the canvas, and then grouped to get linked. Control blocks can be linked in two ways: by dropping at the top side of the current block to add it into the loop, or by dropping at the bottom side to add it after the execution loop.

1) Move instructions: They are used to set the point or path the robot has to execute:

- SETPATH: Once the appropriate block has been dropped in the canvas, if the user presses on it a menu will be shown to select one of the available paths, at the same time that it is displayed in a 3D canvas.
- SETPOINT: It has the same function of SETPATH but in this case a point will be set instead of a path.

2) Robot instructions: They are used to execute robot specific tasks, like deburring, grinding, polishing, etc. At the moment there are two instructions:

- INITIAL POSITION: This block sets the robot initial position, and his default speed. To set these parameters, double click the button and a menu will be shown to write the values.
- DEBURRING: it allows performing deburring operations over a specific path of one of the 3D environment parts. Several parameters can be defined (e.g. depth).

3) Relational operator instructions: They can only be used with the control blocks. They are operations like greater than, lower than, equals or always. To set the parameters, double-click the button and a menu will be shown with the variables available to choose.

4) Variables instructions: Once a variable is created, it can be modified during program execution. Their value can be changed, and they can be added, subtracted, multiplied or divided to other variables or scalars.

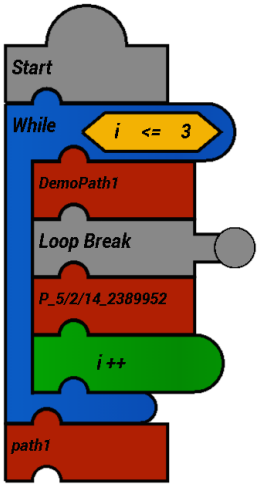

Fig. 3. Program B

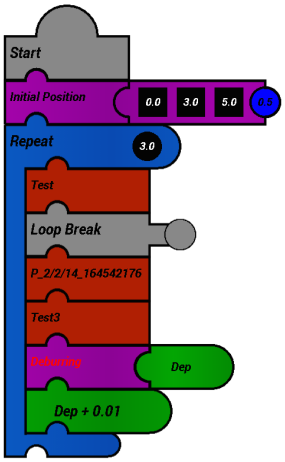

Fig. 4. Program A

5) Control structures: They allow to use control structures like repeat, if, do-while and while to modify the instruction's execution flow. To set the loop execution condition a relational operator block has to be dropped over it. In the repeat case, double click to set the number of times the loop is repeated.

6) User Commands instructions: This group contains instructions that interact with the user.

- START: this is the block that indicates the beginning of the program. If a program doesn't start with this button, it will not be executed.
- LOOP BREAK: this button allows the user to control the execution of a control loop, asking for confirmation to continue the execution loop.

D. Programming examples

Here are some example programs created:

Fig. 3 shows a while loop that is executed while the variable i is lower than 3. Inside the loop a path called demopath1 is executed. In each iteration the user is asked for confirmation to continue the loop and the variable i is increased by 1. Once the while condition is false, the loop ends and the path called path1 is executed.

Fig. 4 shows that the robot initial position has been set at position (0,3,5) with 0.5 m/s speed. Then a repeat loop is executed three times. Inside the loop a path called Test is executed. In each iteration the user is asked for confirmation to continue the loop and then the robot moves to the point P2, executes the Test3 path and does a deburring operation with the depth associated to the variable dep. After that dep will be increased by 0.01.

V. Customized Teach Pendant

Another function that Hammer has implemented is a virtual teach pendant that provides the user with a reduced version of a teach pendant that can be configured depending on the task it is being performed.

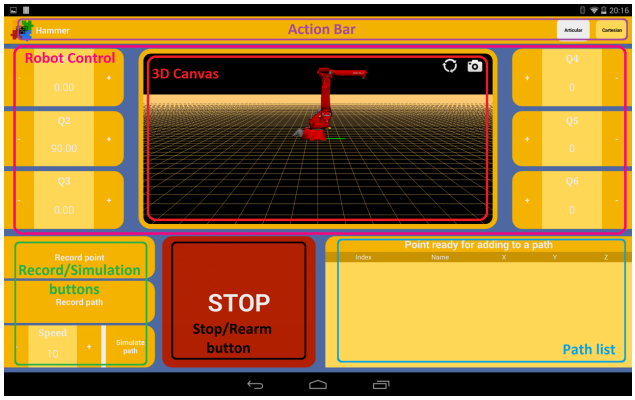

Fig. 5. Customized teach-pendant View

This customized teach pendant includes a simulated robot environment where the operator can save environment points or planning paths without the need of the real robot. It also shows a simulation of the robot executing the programmed task. An image of the interface can be seen in Fig. 5.

The customized teach pendant has 5 main parts:

- Action bar: allows changing the mode from articular to Cartesian coordinates.
- Robot control: depending in the mode articular or Cartesian, the joints or the TCP will be moved. There is a 3D canvas to display the robot movements.
- Record and simulation buttons: These buttons are used to record points (Record Point), paths (Record Path) or execute a simulated path (Simulate Path). A path is recorded by saving a set of points in a row. It is possible to adjust the speed of the simulation pressing the speed button to set the desired speed.
- Stop/Rearm button: If this button is pressed while a simulation is executing, the simulation stops. If it is pressed again, the simulation continues execution.

Fig. 6. Articular Mode

Fig. 7. Cartesian Mode

- Path list: This is a slide list displaying the saved paths and points.

A. *Robot control*

The robot can be controlled in two modes: articular mode and Cartesian mode.

Articular mode offers the possibility to control each of the robot joints through the incremental buttons shown in Fig. 6. In this case the robot is a Comau model with 6 degrees of freedom so it has 6 buttons, one for each joint.

Cartesian mode allows to control the inverse kinematic of the robot so the robot's TCP can be moved along the X, Y and Z axes through one incremental button for each axis (Fig. 7). On the right side there are three buttons to modify the origin of the coordinate system, respect of which the points will be saved.

B. *Points and trajectory generation*

Points and trajectories can be defined using the teach pendant. The robot TCP position can be saved in the points list with an automatically generated name by pressing the save point button. Names can be changed double-clicking the point.

To generate a path, the last points generated are shown in the 3D canvas (Fig.**??**). After pressing the save path button, the path is saved in the path list.

Fig. 8. Auxiliary path generation

Fig. 9. Path 3D representation

When the points and the paths are saved in their respective list a preliminary view of each one is shown when pressing one point or path (Fig. 9).

VI. Augmented Reality Based Monitoring System

Hammer provides a monitoring system based on augmented reality algorithms to display task-specific information in real time. These algorithms have been implemented through the Vuforia library [14].

A Vuforia SDK-based AR application uses the display of the mobile device as a "magic lens" or looking glass into an augmented world where the real and virtual worlds appear to coexist. The application renders the live camera preview image on the display to represent a view of the physical world. Virtual 3D objects are then superimposed on the live camera preview and they appear to be tightly coupled in the real world.

Hammer uses Vuforia to implement local detection of targets and extended tracking, keeping track of targets and maintaining a consistent reference for augmentations even when the targets are no longer visible in the camera view.

As an example, this feature has been used to monitor the robot's force vector. Fig. 10 shows a demo in which the force components of the TCP in X, Y, and Z are displayed, as well as the resulting vector. Values are color-coded (red,green,blue) to represent the magnitude of the applied force. This feature allows the user to observe at all times the force applied to the sensor during the execution of the task the robot is executing.

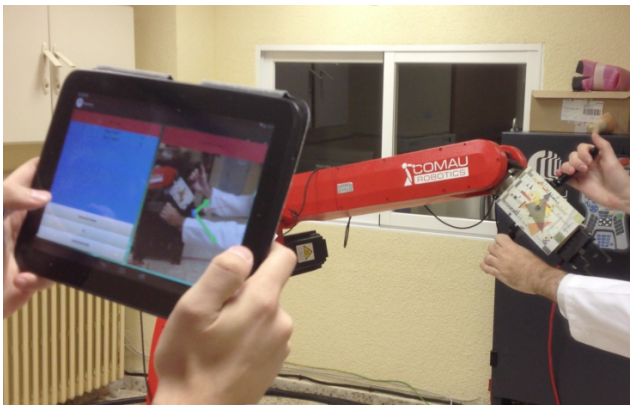

Fig. 10. Augmented reality example: showing force vector

VII. Interface to Other Systems

When the robot program has been finished, the user has three options: execute it in the application simulation environment, execute it in the real robot, or execute it in a third partner simulation software (i.e. EasyRob software [13] is already implemented).

EasyRob is designed to run DLL (dynamic link libraries) from third partners. A DLL has been developed to allow a continuous communication between the Hephestos robot-programming tool and EasyRob software. This DLL implements two main services:

- TCP/IP and UDP/IP socket communications with the android application, to receive and send commands from the application and the computer via Wireless (Wi-Fi).
- Communication between the DLL module and EasyRob, to transmit the commands received from the tool to the robot in EasyRob, and thus visualize the robot movements.

The DLL module has been developed in C++. At the moment it runs only in Windows based PCs.

VIII. Conclusion and Future Work

A new Android based application for industrial robot programming has been presented in this paper. Its main features and functionalities have been described. The application has a robot programming IDE based in visual programming to allow operators with limited programming knowledge to build programs or modified existing ones with ease. The application can also be used to control the robot as a customized teach pendant, and a monitoring tool with augmented reality features.

As this is a first prototype, future work will focus on the development of the programming language (based on Scrach) and the integration with COMAUs robots instruction set (so far a reduce set of instructions is used). In a next step programming by demonstration will be included, allowing the operator to move the real robot with his hand, see the corresponding movement in the application simulation environment, and saving points and paths in the application.

As it has been discussed in the related work section, the use of Android tablets in industrial robot working environments is growing and it is providing a new tool for small batch industrial applications that need fast and easy to use tools to program the robots.

Acknowledgement

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement no 314739 (Hephestos).

References

[1] J. DePree and C. Gesswein, Robotic Machining White Paper Project, Halcyon Development, October 31, 2008.

[2] A. Rogowski, "Robotized cell remote control using voice commands in natural language," *15th Int. Conf. on Methods and Models in Automation and Robotics (MMAR)*. 23-26 Aug. 2010. pp.383,386. DOI: 10.1109/MMAR.2010.5587204.

[3] S. Ganapathyraju, "Hand gesture recognition using convexity hull defects to control an industrial robot," *2013 3rd Int. Conf. on Instrumentation Control and Automation (ICA)*. pp.63,67, 28-30 Aug. 2013. DOI: 10.1109/ICA.2013.6734047.

[4] J. Lambrecht, H. Walzel, and J. Kruger, "Robust finger gesture recognition on handheld devices for spatial programming of industrial robots," *IEEE Int. Workshop on Robots and Human Interactive Communications*. pp.99,106, 26-29 Aug. 2013. DOI: 10.1109/ROMAN.2013.6628462.

[5] Slany, W., "A mobile visual programming system for Android smartphones and tablets," 2012 *IEEE Symp. on Visual Languages and Human-Centric Computing (VL/HCC)* pp.265,266, Sept. 30 - Oct. 4 2012. DOI: 10.1109/VLHCC.2012.6344546.

[6] S. Van Delden and A. Whigham, "A bluetooth-based architecture for android communication with an articulated robot," *Int. Conf. on Collaboration Technologies and Systems (CTS)*. 2012. pp.104,108, 21-25 May 2012. DOI: 10.1109/CTS.2012.6261035.

[7] J.C. Yepes et al., "Implementation of an Android based teleoperation application for controlling a KUKA-KR6 robot by using sensor fusion," *Health Care Exchanges (PAHCE)*, 2013 Pan American, pp.1,5, April 29 2013-May 4 2013 DOI: 10.1109/PAHCE.2013.6568286.

[8] O. Neira, A.N. Lee, J.L.M. Lastra and R.S. Camp. "A builder for Adaptable Human Machine Interfaces for mobile devices," *11th IEEE International Conf. on Industrial Informatics (INDIN)*. pp.750,755. 29-31 July 2013. DOI: 10.1109/INDIN.2013.6622978.

[9] M. Nicolae, L. Lucaci and I. Moise. "Embedding Android devices in automation systems," *IEEE 19th Int. Symp. for Design and Technology in Electronic Packaging (SIITME)*. pp.215,218, 24-27 Oct. 2013. DOI: 10.1109/SIITME.2013.6743676.

[10] Y. Jan, S. Hassan, S. Pyo and J. Yoon, "Smartphone Based Control Architecture of Teaching Pendant for Industrial Manipulators," *2013 4th Int. Conf. on Intelligent Systems Modelling and Simulation (ISMS)*. pp.370,375, 29-31 Jan. 2013. DOI: 10.1109/ISMS.2013.116

[11] S. M. Abbas, S. Hassan and J. Yun, "Augmented reality based teaching pendant for industrial robot," Control, 2012 12th Int. Conf. on Automation and Systems (ICCAS). pp.2210,2213, 17-21 Oct. 2012.

[12] Hephestos EU FP7 project. Available: http://www.hephestosproject.eu/. [Accessed: July 07, 2014].

[13] EasyRob simulation software. Available: http://www.easy-rob.com/. [Accessed: July 07, 2014].

[14] Vuforia library. Available: http://www.qualcomm.com/solutions/augmented-reality. [Accessed: July 07, 2014].

[15] Scratch programming IDE. Available: http://scratch.mit.edu/.[Accessed: July 07, 2014].