Accurate manual guided robot programming and trajectory correction using 3D vision by laser triangulation

Alberto Tellaeche, Ramón Arana, Miguel Ángel Pérez, Iñaki Maurtua

***Abstract*— Normal utilization of robot manipulators of anthropomorphic type does not reach beyond reiteration of pre-programmed trajectories. While static robot programs may be sufficient for high volume manufacturers, they are not adequate in one-off or small-batch manufacturing. The objective of the research presented in this paper is to facilitate robot programming by combining hand guided end-effector rough movement planning and 3D visual servoing based accurate trajectory following.**

I. Introduction

Nowadays, one of the main bounds for the growth and widespread of robotized cells in the context of small and medium enterprises is the complexity in the programming of robots. In a large number of industrial realities the training level for that kind of operation represents one of the biggest obstacles in order to prefer stiffer and more limited automation solutions, intrinsically easier to setup.

This is particularly true for SMEs that cannot afford for big investments required for robot introduction and use, and cannot make expensive efforts in personnel robot training. This last aspect, cost of use, is most relevant when we think on small or even unitary productions (hyper-flexible cell scenario) where programming is too much time demanding.

The objective of this research is to integrate a breakthrough programming approach combining a universal Manual Guidance Device (MGD) for a fast, intuitive but rough tool path programming with a 3D visual servoing approach to adjust the obtained trajectories and to allow accurate end-effector positioning by automatic correction of Tool Central Point (TCP) path. The user decides the distance between the end-effector and the part on top of which the application has to process (welding, painting, deburring, etc.), the orientation with respect the part, the type of correction (follow the edge, follow the contour, etc.) and any other parameter relevant for the process.

This research intends to make possible to program robotic applications in an easy way by not experts in robotics that can focus their attention on the process (laser cladding, deburring, ...) regardless programmatic aspects.

II. Related Work

During the last years there have been several approaches related with the programming of robots by manual guidance:

In [1], Frigola et al. the programming of industrial robots with the methods of 'Programming by Manual Guidance' is described.

Ang et al. [2] describe a programming system for welding in shipyards. Robots can be programmed in a very fast pace by a 'walk-through approach'.

Leeser et al. [3] implemented another programming system called 'Computer- Assisted Teach and Play' for the whole arm manipulator of Barrett Technology Inc.

Other interesting methods and strategies for this type of robot programming can be found in [4], [5] and [6].

In Bruyninckx et al. [7] an approach for real-time robot programming by human demonstration for 6D force controlled actions has been done.

Finally, Sato et al. [8] present an alternative method of robot programming that does not need the use of force/torque sensors.

Regarding 3D machine vision for surface control and coordinate extraction, works can be found at [9] and [10].

III. Manual Robot Programming

In this research, a COMAU NM45 robot has been used. In 2008 COMAU Robotics leaded the development of a low cost programming by demonstration device based on an optical mouse with 6 degrees of freedom and an embedded system capable of acquiring and computing signals, an also interfacing via Bluetooth (or WiFi) with the COMAU Wireless Teach Pendant. This MGD device will be used to program the robot trajectory in an intuitive but rough way.

Fig. 1: MGD and teach pendant (COMAU)

Parallel to trajectory program, the robot executes a parallel task to register the TCP positions in a text file. These positions are recorded whenever an increment of 0.5mm is detected in the Euclidean distance from the previous recorded point or when an increment of 1º is detected in the α angle (rotation of the camera). The registered points have the following formal definition:

$$< x, y, z, \alpha, \beta, \gamma > \quad (1)$$

A. Tellaeche et al. are with Fundación Tekniker, Apdo. 44 Otaola 20, 20600 Eibar, Gipuzkoa, Spain (phone: +34 943 20 67 44; fax: +34 943 20 27 57; e-mail: atellaeche@tekniker.es, rarana@tekniker.es, maperez@tekniker.es, imaurtua@tekniker.es).

IV. 3D Image Acquisition

For 3D visual servoing and trajectory correction, a laser triangulation system has been used. This system is composed by a SICK Ranger E55 Camera and a Class 2, red laser line. The camera has a 1536 * 512 pixel sensor and is capable of obtaining up to 35K profiles per second.

Next figure shows a schematic representation of the geometrical setup for laser triangulation. According to [11], this *ordinary setup* provides the maximum height resolution.

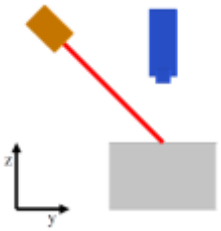


Fig. 2: Ordinary setup of the laser/camera for maximum height resolution in 3D laser triangulation

The Ranger E55 camera internally calibrates the 3D images obtained, giving the *d* and *z* coordinates in real world mm. *d* is the Euclidean distance from the target (borders) to the center of the camera sensor, and *z* the point height.

This triangulation system is mounted in the robot flange with the MGD. Using the 3D calibrated images acquired with the recorded coordinate points of the robot it is possible to correct the rough trajectory recorded by the MGD, obtaining the very accurate path necessary in many robotic applications.

In fig. 2, the COMAU robot flange with both systems can be observed.

Fig. 3: COMAU robot flange with MGD and 3D triangulation system

V. Experiment Definition

Based on the needs of the experiment, a prototype board has been constructed in Aluminum. This prototype has different trajectories defined, such as: straight line, curve, spline, etc. The track defined in the prototype has a width of 60 mm and a depth of 7 mm, with a central nerve of 10 mm. Detail of this board can be observed in fig. 2.

The overall system is moved by the MGD, and while the rough trajectory is being defined, coordinate points are recorded as explained in Section III. The robot synchronizes the recording of each point with a 3D line scan of the camera by a trigger pulse.

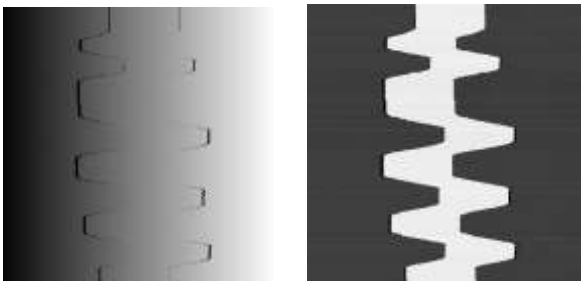
Every 500 pulses, the file containing the 500 coordinates and the 1536 * 500 pixel 3D image acquired containing *d* and *z* calibrated coordinates are processed for trajectory correction. In parallel the next capture of points is being performed.

VI. Trajectory Correction and Adjustment

Prior to robot movement, the user must select what type of edge must be adjusted in the subsequent operation. Options are, in order: left border of the track, left border or the central nerve, right border of the central nerve, right border of the track. This setup will be used in the image processing to detect the correct border. Also, the reference *z* coordinate must be adjusted.

A. Image Processing

As explained in previous sections every 500 recorded points, two images are obtained from the ranger camera, *d* calibration and *z* calibration.



(a) *(b)*

Fig. 4: (a) *d* coordinate calibration, (b) *z* coordinate calibration

After performing image filtering for noise reduction, the line profiles composing the 3D *z* calibrated image are processed, calculating the derivate signal for each one. With the derivative signal, it is possible to detect in a very precise manner the level transitions (borders) present in the line profile. When there is a high to low transition, corresponding to the left track border or to the right border of the central nerve, a minimum peak appears in that point in the derivative signal. On the other hand, when there is a low to high transition corresponding to left border of the central nerve, or to right track border, a maximum appears in the derivative signal. The rest of the values in the derivative signal tend to 0.

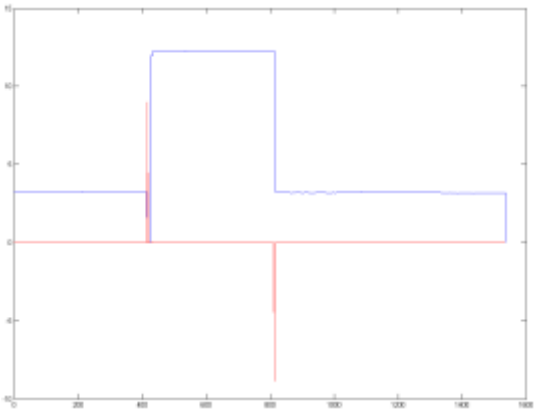


Fig. 5: Line profile (blue) and its derivative signal (red)

Taking in to account the maximums and minimums present in the derivative signal and their relative position, it is possible to estimate the pixel in the line profile where a certain border is. With that pixel value, the *z* calibrated value is obtained from the *z* calibrated image, and the *d* calibrated value is obtained from the *d* calibrated image. These values are converted to robot coordinates in the format presented in (1). This conversion is:

$$x = d\ cos(\alpha) \quad (2)$$

$$y = d\ sen(\alpha) \quad (3)$$

B. Trajectory estimation

By the process explained in the previous section, it is possible to estimate the real points of the border being inspected and thus, obtain the trajectory. Although the image is processed to avoid errors due to noise or lack of data in certain points, it has been implemented a restriction in the position of the points in the trajectory, taking into account the previous (*x*,*y*) positions of the two previous correct points obtained.

Let (x_0, y_0) and (x_1, y_1) be the two previous correct points calculated in the trajectory. The maximum increment in *x* and *y* coordinates is defined by a maximum Euclidean distance of 0.5 mm between two consecutive points.

$$(x_{2max}, y_{2max}) = (x_1 + \Delta x_{max}, y_1 + \Delta y_{max}) \quad (4)$$

If $x_2 > x_1 + \Delta x_{max}$ or $y_2 > y_1 + \Delta y_{max}$, then (x_2, y_2) is estimated as follows:

$$y_2 = y_1 + \Delta y_{max} \quad (5)$$

$$x_2 = (x_1 - x_0)(y_2 - y_0) / (y_1 - y_0) + x_0 \quad (6)$$

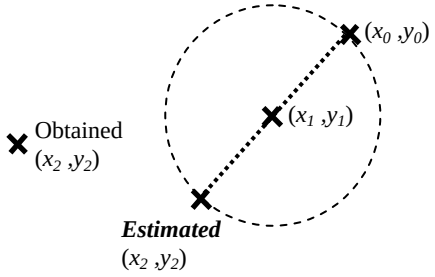


Fig. 6: Graphical representation of faulty point estimation

C. Trajectory correction

During system set up, the center of the sensor of the Ranger E55 camera has been adjusted and calibrated to coincide with the robot TCP. Thus, the x_c and y_c point coordinates of the border detected in the 3D image correspond to the Δx and Δy final corrections of each point recorded by the robot.

Given a recorded point of the robot:

$$X_r = < x_r, y_r, z_r, \alpha_r, \beta_r, \gamma_r > \quad (7)$$

and the coordinates of the border obtained from the camera:

$$X_c = < x_c, y_c, z_c, \alpha_c, \beta_c, \gamma_c > \quad (8)$$

The final correction is:

$$X_r - X_c = < x_r - x_c, y_r - y_c, z_r, \alpha_r, \beta_r, \gamma_r > \quad (9)$$

The α angle variation is implicitly taken in to account in (2) and (3), so it is not necessary to change the recorded value from the robot. *z* coordinate is calibrated and the real height value of the point of the trajectory being followed is obtained from the camera.

VII. Experimental Results

In order to assess the performance of the correction method proposed, several tests have been carried out, using three different types of trajectories and two different borders in the prototype.

The three trajectories used for validation are: straight, curve and combined straight-curve trajectories.

The borders selected for testing are left border or the central nerve and the right border of the track.

Table 1 shows the results obtained. The precision of the trajectory corrected is expressed in terms of mean and standard deviation of the accumulated error in trajectory estimation. These measurements are expressed in mm.

TABLE I. TRAJECTORY ESTIMATION RESULTS

Predefined trajectories	Left border of the central nerve		Right border of the track	
	Mean	**Std. Dev.**	**Mean**	**Std. Dev**
Straight	0.013	0.043	0.014	0.037
Curve	0.068	0.091	0.072	0.102
Straight - Curve	0.057	0.077	0.051	0.069

a. All measurements are in mm.

Discussing the results obtained, it can clearly be seen that the system has a better performance in correcting straight trajectories. This is caused because in curve trajectories, there are more variables that can increment the overall error. While in straight trajectories all the inaccuracies depend purely on linear movements, in curve trajectories, apart from these linear movements, there are also flange rotations. However, the errors obtained can be perfectly assumed in the majority of robotic applications.

VIII. CONCLUSIONS AND FUTURE WORK

With this research, it has been developed a new system for quick robot programming. The rough trajectory generated with the MGD, can afterwards be precisely corrected with the calibrated data obtained from the 3D images provided by the Ranger E55 camera.

The work done until now corrects the trajectories but does not take into account the robot tool orientation (β and γ angles). Tool correction and orientation will be the next steps for this research to obtain total robot trajectory correction.

IX. ACKNOWLEDGMENT

This research has been carried out within the scope of the experiment “EASYPRO – Accurate Manual Guided robot programming”. EASYPRO experiment belongs to the ECHORD (European Clearing House for Open Robotics Development) platform, an EU-funded project within the Seventh Framework Program, aiming to strengthen the cooperation between scientific research and industry in European robotics.

REFERENCES

[1] M. Frigola, J. Poyatos, A. Casals, J. Amat, "Improving Robot Programming Flexibility through Physical Human - Robot Interaction" *IROS Workshop on Robot Programming by Demonstration*, Las Vegas, October 2003.

[2] H. Ang, W. Lin, S-Y Lim, "A Walk-Through Programmed Robot for Welding in Shipyards", in *Industrial Robots*, Vol. 26, No. 5, 1999.

[3] K. Leeser, J. Donoghue, W. Townsend, "Computer-Assisted Teach and Play: Novel User-Friendly Robot Teach Mode Using Gravity Compensation and Backdrivability", in *Proceedings of S ME Fifth World Conference on Robotics Research*, Cambridge, USA, 1994.

[4] A. Albu-Schaeffer, G. Hirzinger, "Cartesian Impedance Control Techniques for Torque Controlled Light-Weight Robots", in *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, Washington DC, May 2002.

[5] G. Grunwald, G. Schreiber, A. Albu-Schaffer, G. Hirzinger, "Programming by touch: The different way of Human-Robot Interaction", in *IEEE transaction on industrial electronics*, Vol. 50, No. 4, 2003.

[6] H. Asada, S. Izumi, "Direct Teaching and Automatic Program Generation for the Hybrid Control of Robot Manipulators", in *IEEE International Conference on Robotics and Automation*, Raleigh, 1987.

[7] H. Bruyninckx, J. De Schutter, "Specification of force controlled actions in the task frame formalism — a synthesis", in *IEEE Trans. on Robotics and Automation*, vol. 12, no. 4, pp. 581-589.

[8] D. Sato, T. Shitashimizu, M. Uchyama, "Task Teaching to a Force Controlled High Speed Parallel Robot", in *Proceedings of International Conference on Robotics & Automation*, Taipei, Taiwan, 2003.

[9] J. Leopold, H. Gunther, R. Leopold, “New developments in fast 3D-surface quality control” in *Measurement*. 2003, 33(2), 179-187.

[10] A. Picón, M.A. Bereciartua, J.A. Gutiérrez, J. Pérez. “Machine vision in quality control. Development of 3D robotized laser-scanner” in *Dyna*. 2010, 84(9), 733-742.

[11] K.E. Boehnke, *“Hierarchical Object Localization for Robotic Bin Picking”*. Ph.D. dissertation. Faculty of Electronics and Telecommunications. Politehnica University of Timisoara. September 2008.

Assistive Robots for Grit-Blasting in SteelBridge Maintenance: Technological Progress and Innovation

D.K. Liu, *Member, IEEE*, G. Dissanayake, *Member, IEEE*, P.B. Manamperi, P.A. Brooks, W. Kaluarachchi

***Abstract*—Application of robots in complex steel bridge structural environments is challenging. There are a number of research and engineering challenges that need to be addressed. This paper presents the technological progress and innovation of an industry and academia research collaboration project "A robotic system for steel bridge maintenance". This project has been jointly funded by industry, government and university since 2006. A number of research challenges have been addressed and two prototype robots developed. Three field trials have been successfully conducted in the iconic SydneyHarbourBridge. This project is now at the stage of commercial-ready product development with commercialisation activities started.**

I. Introduction and Brief Project History

Bridges are critical links in the transport network benefiting communities and facilitating the growth of economy. There are approximately 42,000 steel bridges in Europe, and 210,000 and 270,000 steel bridges, respectively in the USA and in Japan[1]. Corrosion is the primary cause of failure in steel bridges, which is minimized by coating. The paint requires periodic inspection and maintenance. Inadequate maintenance may result in, or contribute to structural failures such as the MississippiBridge incident in Minneapolis that lead to 13 fatalities, 145 injuries [2] and a replacement cost of several hundred million dollars.

Periodic inspection and maintenance of steel bridges are vital but very expensive undertaking because of the associated employee health and safety issues. Steel bridges such as the SydneyHarbourBridge are normally very complex structures.Grit-blasting to remove rust and old protective coatings followed by re-painting is the common approach in steel bridge maintenance, but grit-blasting is extremely labour intensive and hazardous. Thus supplementing manual labour in steel bridge maintenance with robotic aids has significant safety, cost and health impacts.

Application of robots in complex steel bridge environments is challenging. There are many research and engineering questions that need to be answered. In 2005, the Centre for Autonomous Systems at the University of Technology, Sydney (UTS) and the Roads and Maritime Services (RMS) of New South Wales in Australia initiated and started a collaborative research project of "A robotic system for steel bridge maintenance _Stage I". This project was first funded by UTS and RMS under the UTS Partnership Grant, with the focuses on scoping study, proof of concepts and platform development.

The second stage of this project was jointed funded by RMS, the Australian government and UTS from 2007-2010. Three post-doctoral research fellows, five PhD students, five engineers, five bridge maintenance workers and over 15 final year undergraduate students worked for this project. Systematic study on enabling methodologies was conducted with solutions to challenging research issues developed. A prototype robotic system was developed and tested in both laboratory setup environment and on-site bridge maintenance environments. The theoretical research outcomes have been implemented in the prototype robot and verified in the field trial.

With significant support from both RMS and UTS, the project team started in 2011 to develop two workable robotic systems for extensive testing and long-term use. A new prototype robotic system has been developed with its performance significantly improved in terms of functionality, reliability, operation interface. This new prototype robot has been tested twice in bridge maintenance sites.

II. Technological Progress and Innovations

The robotic system consists of a 6-DOF industrial robot, a moving platform, a sensor package and computers (Figure 1) [3]. An industrial robot with a payload of over 10kg is needed in order to handle the grit-blasting nozzle reaction force (over 80N [4]). The whole system is placed on the floor of a scaffold which is fully enclosed.

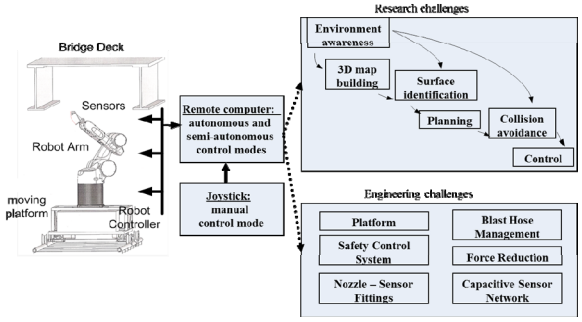


Figure 1. The steel bridge maintenance robotic system and its functional modules

*Resrach supported by the Roads and Maritime Services, New South Wales; the University of Technology, Sydney and the Australian Research Council.

D.K. Liu and G. Dissanayake are with the Centre for Autonomous Systems, University of Technology, Sydney, Australia, POBox 123, Broadway, NSW 2007. (e-mails: Dikai.liu@uts.edu.au; Gamini.dissanayake@uts.edu.au).

P.B. Manamperi, P.A. Brooks and W. Kaluarachchi are with the Roads and Maritime Services, New South Wales, Australia. (e-mails: {palitha.manamperi, philip.brooks, waruna.kaluarachchi}@rms.nsw.gov.au)

For achieving autonomous operation of the robot in unknown and complex 3D environments, the research challenges that need to be addressed including sensing, exploration, 3D mapping of a complex bridge structural environment, surface classification, identification of material types in a bridge structure, real-time grit-blasting planning, on-line robot trajectory planning, and collision detection.

(1) Sensing. Sensors such as laser range finder and RGB-D camera can be used for sensing and map building of an environment. In our research, Hokuyo laser range finders and RGB-D camera have been used because they are light in weight and small in size. Installing the sensor onto the end effector of the robot manipulator gives the robot the capacity of exploring complex 3D environments, but one important issue related is protection of the sensors from damage that can be caused by the rebounded high speed abrasive grit during grit-blasting. Sensing of material types in a steel bridge environment is a challenge because all members in the environment may be painted with the same colour, and high-speed grit will damage members made of wood or plastic materials. Obstacle ranging in particle laden air is another challenge. Capacitive sensing and sensors [5][6] has been studied and developed for material type classification and obstacle ranging during blasting.

(2) Exploration and 3D map building. For addressing the challenge of exploring a complex steel bridge environment and building the 3D map of the environment, an sampling based exploration approach [7], a simultaneous mapping and surface identification method [8] and a surface growing algorithm [9] have been developed in this project and implemented in the prototype robotic system.

(3) Real-time on-line grit-blasting planning. Steel bridge maintenance requires the robot to move from section to section after one section of bridge structure is grit blasted. Therefore the environment is keeping "changing" from the robot point of view. Productivity expectation requires the robot to conduct grit-blasting very efficiently and with minimum setup time and maximum surface coverage. Therefore, efficient real-time grit blasting planning and robot trajectory planning is critical. This research developed effective real-time planning algorithm from grit-blasting and robot trajectory planning [10].

(4) Real-time collision detection and avoidance. Due to the complexity of the environment, collision detection and avoidance is a critical issue, particularly at the exploration and map building stage because the environment is unknown to the robot. Expected productivity requires the robot to move fast. A 3D force field method [11] has been developed for effective real-time collision detection and avoidance.

(5) User collaborative design. Different from the approach of "Design for User", a user collaborative design approach is applied in this research. The approach involves end-users in the design team to work together with the researchers. It has been proved that the usability and acceptance of the robotic system has been significantly improved.

Besides research challenges, there are many significant engineering issues as shown in Figure 1 due to the very specific application. Solutions have been developed for addressing these engineering issues [12]. Two robotic grit-blasting systems have been developed (Figure 2). Three field trials have been successfully conducted[13], which validated the robotic system and the developed algorithms and methods.

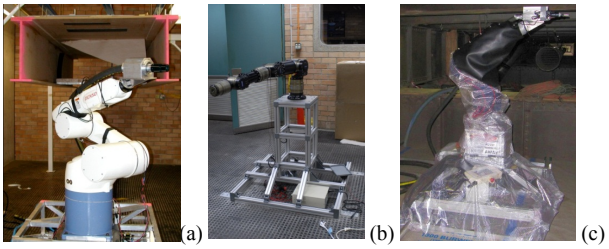

(a) (b) (c)

Figure 2. Prototype grit-blasting robotic systems: (a) prototype 1 with Denso robot arm; (b) prototype 2 with Schunk joints; (c) prototype 2 with protection cover

ACKNOWLEDGMENT

The authors thank all the team members who made contributions to the project.

REFERENCES

[1] C. Balaguer, A. Gimenez, & A. Jardon (2005), "Climbing Robots' Mobility for Inspection and Maintenance of 3D Complex Environments", Autonomous Robots, vol. 18, no. 2, 2005, pp. 157-169.

[2] C.R. Schultheisz, A.S. Kushner, et al , "Minneapolis I-35W Bridge Collapse – Engineering Evaluations and Finite Element Analysis", Posted in Civil Engineering Disasters, http://www.engineeringcivil.com/minneapolis-i-35w-bridge-collapse-engineering-evaluations-and-finite-element-analysis.html#more-3110

[3] D.K. Liu, G. Dissanayake, P.B. Manamperi, P.A. Brooks, G. Fang, G. Paul, S. Webb, N. Kirchner, P. Chotiprayanakul, N. M. Kwok, T.R. Ren (2008), "A robotic system for steel bridge maintenance: research challenges and system design", Proceedings of the Australasian Conference on Robotics and Automation (ACRA 08), 3-5 December 2008, Canberra, Australia, 7 pages.

[4] N. Kirchner, G. Paul, D.K. Liu (2006), "Bridge maintenance robotic arm: mechanical technique to reduce the nozzle force of a sandblasting rig", Proceedings of the 1st International Symposium on Digital Manufacturing, October 15-17, 2006, Wuhan, China, (also included in the Journal of Wuhan University of Technology, Vol. 28, Suppl.1), pp12-18

[5] N. Kirchner, D.K. Liu, G. Dissanayake (2006), "Bridge maintenance robotic arm: capacitive sensor for obstacle ranging in particle laden air", Proceedings of the 23rdInternational Symposium on Automation and Robotics in Construction 2006 (ISARC 2006), October 3-5, 2006, Tokyo, Japan, pp596-601

[6] N. Kirchner, D.K. Liu, T. Taha and G. Paul (2007), "Capacitive object ranging and material type classifying sensor", Proceedings of the 8th International Conference on Intelligent Technologies (InTech), 12-14 December 2007, Sydney, Australia, pp130-135.

[7] G. Paul, S. Webb, D.K. Liu and G. Dissanayake (2011), "Autonomous

Robot Manipulator-based Exploration and Mapping System for Bridge Maintenance", Robotics and Autonomous Systems, Vol. 59, Issues 7-8, July-August, pp543-554

[8] G. Paul, D.K. Liu, N. Kirchner and G. Dissanayake (2009), "An effective approach to simultaneous mapping and surface-type identification of complex 3D environments", *Journal of Field Robotics*, 26 (11-12): pp915-933, November-December 2009

[9] G. Paul, D.K. Liu, N. Kirchner(2007), "An algorithm for surface growing from laser scan generated point clouds", in T.-J. Tarn, S.B. Chen and C. Zhou (Eds), "Robotic Welding, Intelligence and Automation", Lecture Notes in Control and Information Sciences (Vol. 362), Springer-Verlag, 2007, pp481-491

[10] T.R. Ren, N. M. Kwok, D.K. Liu and S.D. Huang (2008), "Path planning for a robotic arm sand-blasting system", Proceedings of the IEEE International Conference on Information and Automation, June 20-23, 2008, Zhangjiajie City, Hunan, China, pp1067-1072

[11] P. Chotiprayanakul, D.K. Liu, D. Wang, G. Dissanayake (2007), "A 3-dimensional force field method for robot collision avoidance in complex environments", Proceedings of the 24th International Symposium on Automation and Robotics in Construction (ISARC 2007), 19-21 September 2007, India, pp139-145

[12] P.B. Manamperi, P.A. Brooks, W. Kaluarachchi, G. Peters, A. Ho, S. Lie, A. To, G. Paul, D. Rushton-Smith, S. Webb, D.K. Liu, G. Dissanayake (2011), "Robotic Grit-blasting: Engineering Challenges", Proceedings of the 8th Austroads Bridge Conference (ABC 2011), 31 October to 5 November 2011, Sydney, Australia, pp321-330

[13] G. Paul, S. Webb, D.K. Liu, G. Dissanayake (2010), "A Robotic System for Steel Bridge Maintenance: Field Testing", Proceedings of the 2010 Australasian Conference on Robotics and Automation (ACRA 2010), 8 pages, 1-3 December, 2010, Brisbane, Australia

Grab a Mug – Grasp Motion Planning with the Nao Robot

Judith Müller
Deutsches Forschungszentrum für Künstliche Intelligenz, Cyber Physical Systems,
Enrique-Schmidt-Str. 5, 28359 Bremen, Germany
Email: judith.mueller@dfki.de

***Abstract*— Robots such as the humanoid Nao [1] manufactured by Aldebaran Robotics demonstrate that versatility does not always has to be expensive. Although Nao's motorization is less strong compared to skilled robots such as Justin [2] or ASIMO [3] its performance in tasks like playing soccer [4] is convincing. Nevertheless tasks such as grasping constitute as particular problem due to the limited processing power and the hand design.**

In a joint project between the manufacturer Aldebaran Robotics and the academic research institute DFKI, a grasp function has been developed for the Nao robot. It plans a hand motion path using a pre-calculated workspace while avoiding obstacles in order to grasp known objects, thereby allowing to grasp objects in real-time on an affordable humanoid robot.

I. Introduction

Humans are able to interact with their environment. Besides countenance, speech, and gestures, humans can also manipulate the surrounding with their hands in order to do tasks such as object grasping. Thus, humans are able to do actions such as lifting up a coffee cup in order to drink. Since hands are one of the most useful human features, it seems that robots also could benefit.

If for instance the domestic service robot Roomba had hands [5] he could pick up small objects from its vacuum route in order to clean more efficient. With hands it also would be capable of doing more complex tasks such as putting the trash out or move the cloth into the washer. But at most these tasks are executed by expensive research prototypes that only act in special environments.

In this paper we present a grasp function for the affordable Nao robot [1]. We focused on grasping objects in real-time despite the limited processing power. In contrast to [6] the proposed grasping function plans and executes only single handed grasps using an A* based algorithm. The object to grasp is a standard-sized coffee cup.

There are many different ways to grasp an object, but not every option is the best. In [7] grasps are distinguished in force-closure and form-closure grasps. Force-closure grasps are able to balance disturbances by the wrenches (fingers) at the contact points. In contradiction to that are form-closure grasps, whereby no friction is on the contact points, so that disturbances can be ignored.

Nao's hands are underactuated [8] because they are constructed with three flexible fingers per hand, which are controlled by a single motor. Unfortunately, the fingers are only really stiff if the hand is completely closed. Hence, experiments showed that solid objects such as coffee cups are only graspable if the grasp is form-closure. Furthermore, it seems that performing a force-closure grasp is not possible with this robot, since it is not able to move its fingers individually [9]. The grasping hand is attached to a humanoid robot, which can move in order to reach certain objects. For that reason it is necessary to validate whether the object to grasp is reachable. This can be a very expensive process, because the reachability needs to be checked by inverse kinematics for many points in order to select a suitable grasp. Also a humanoid robot has different movable body parts (i. e. legs) that need to be checked for collisions and therefore need to be avoided.

In order to speed up the planning process we defined the robot's workspace based on the capability map introduced in [10] and [11]. By defining the workspace and storing information about how a region can be reached by the hand, it is possible to plan a grasp path without the direct use of inverse kinematics.

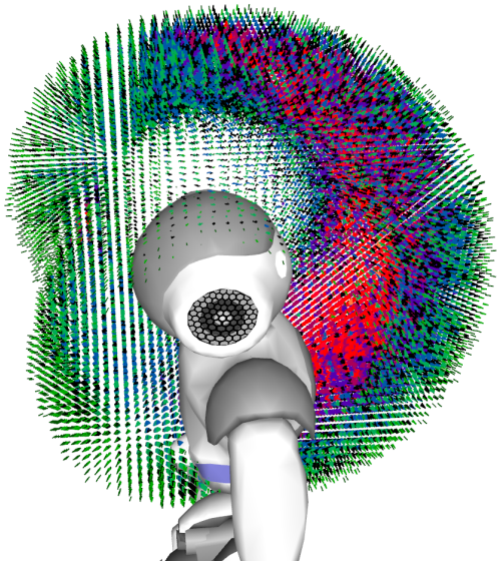


Fig. 1. The best reachable regions are marked as red (14 reachable lower arm rotations), less good reachable regions are marked blue (7 reachable lower arm rotations) and bad regions are marked green (1-2 reachable lower arm rotations).

This work has been funded by the European Commission in the 7th Framework Programme for RTD in the context of the project 'ECHORD – European Clearing House for Open Robotics Development" under the contract number FP7-231143.

II. Reachability Map

In our approach we concealed the workspace with a cube that is divided into equally sized smaller cubes. Each subcube serves as a region in the workspace. Each region stores a set of reachable directions, which indicate the lower arm rotations possible. Fig. 1 pictures the reachability map used, where only reachable directions per region are marked.
Because the Nao has a 5DOF arm, we use the lower arm rotation in order to represent the hand rotation. With that approach the definition of the workspace becomes more evident, because the hand position depends on the lower arm rotation. Therefore 4 DOF are used to represent a hand pose and the 5th DOF is used to represent the rotation around the lower arm.
The map is created offline by using forward kinematics for each 0.5 degree angle of each arm servo. In that process we calculate the position of the palm as well as the rotation of the lower arm and mark them in the map. In order to keep the memory size as small as possible, each lower arm rotation calculated is matched with a set of 512 direction, which are generated by using the spiral point algorithm proposed by Saff and Kuijlaars [12].
The origin of the reachability map is in the shoulder of the robot. Therefore, it is possible to test with different shoulder positions if a certain hand position is reachable without the use of inverse kinematics. Fig. 2 pictures how the workspace increases with only four possible shoulder positions.

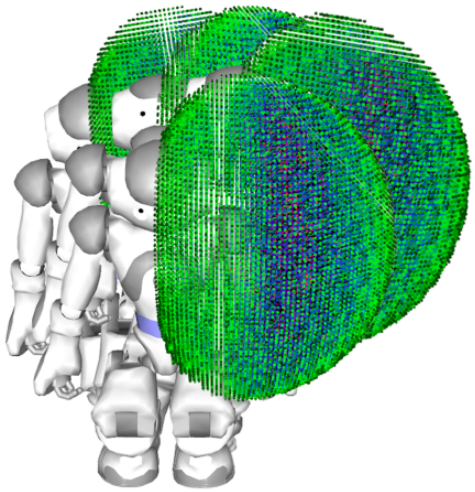


Fig. 2. Extended reachability map due to more possible shoulder positions: sit down, lean left, lean right, and stand

III. Planning

The goal of motion planning is to find a valid path from the current hand position to a position where the object can be grasped. In addition, obstacles such as the object itself or body parts that may be in the direct path between the target position and the start position should be avoided.

A. Grasp Selection

Before the motion plan can be constructed, a target position needs to be selected. In order to do this, we assigned a set of predefined grasp rules to each object. Each rule is defined by a grasping point and a range of lower arm rotations. In Fig. 3, grasp rules are marked with blue triangles. The green dots constitute the position where to grasp and the triangle defines, in which range the lower arm should be rotated.
In order to select a grasp the grasp rules are matched with the reachability map defined in Sec. II. In this process areas that include a grasping point are further examined in order to check if the corresponding possible lower arm rotations are qualified for the grasp. This step is necessary because the possible lower arm rotations per area are very limited (Fig. 1). In this process the possible lower arm rotations (red in Fig. 3) of the grasp areas are compared with the angle ranges from the grasp rules (matching rotations are yellow in Fig. 3) whereas the best match is selected.

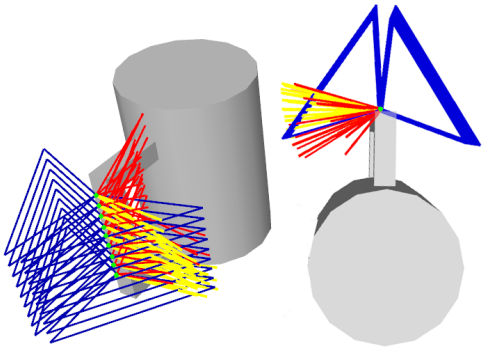


Fig. 3. Each object has its own grasp map, which is generated by a set of predefined grasp rules. Each rule connects a range of lower arm rotations to a grasp position (blue). The reachability map (red) is matched to the object's grasp map, matches are marked yellow.

B. Motion Planning

The next step is the actual planning from the selected grasp position (start node) to the current hand position (goal node). The planning algorithm calculates directly on the map data without reachability checks via inverse kinematics.
The reachability map provides the planner with 6D information of the possible hand positions and lower arm rotations. Since planning in 6D is very expensive, our A* based planning algorithm initially uses only the 3D area grid and ignores the lower arm rotation. Thereby, to be evaluated, nodes are checked on reachability and obstacle overlay in order to calculate heuristics only for verified nodes. In this process, nodes with more suitable lower arm rotations are rated better than nodes with greater deviations from the lower arm goal rotation. Also the distance between the node evaluated and the goal node in 3D are taken into account.
The output from the planning algorithm is a list of way points through the reachability map, which are represented as red dots in Fig. 4. Since the hand positions are in dependence with the rotation of the lower arm, a way point also includes

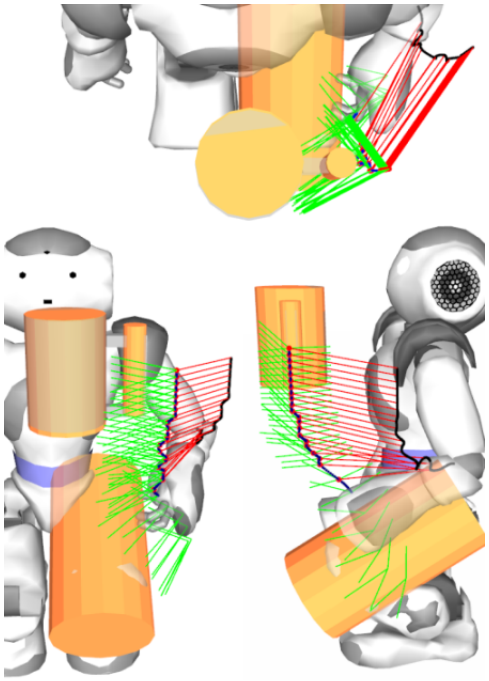


Fig. 4. The interpolated motion path of the hand and the elbow are marked blue and black. Lower arm rotations are represented with red lines, which are always connected to a way point (red dots). Obstacles are marked by orange cylinders and the finger positions calculated per way point are represented by green lines.

a rotation. Each rotation declares one elbow position per way point and is marked by red lines in Fig. 4. In order to execute a plan found, a trajectory-based motion engine [13] was extended to take arm movements as input. Since the grasp plan consists of waypoints, it contains uncontrollable velocities in the connecting points. In order to overcome this problem, the grasp plan is converted into a Bezier spline using the method by DeRose *et al.* [14]. This Bezier spline is marked blue in Fig. 4.

C. Heuristic

In equation (1), the heuristic h_σ for a node x is calculated from five parameterized equations (2) with $p_{[0...4]}$ as factors. The function $l(x)$ in equations (2) calculates the corresponding elbow position for a node x and the function $a(x)$ calculates the corresponding arm angle for a node x.

$$h_\sigma = \sum_{k=0}^{4} p_k h_k \tag{1}$$

$$h_0 = |x - x_{goal}|; h_1 = |l(x) - l(x_{goal})|;$$
$$h_2 = |l(x) - l(x_{last})|; h_3 = |a(x) - a(x_{goal})|$$
$$h_4 = |a(x) - a(x_{last})| \tag{2}$$

The equations (2) evaluate, how the next node x connects to the goal node and the previous node. This is necessary, because most nodes are only reachable by a few possible lower arm rotations as it is shown in Fig. 1. If the heuristic would only take the distance into account then there could be nodes on the path where the lower arm rotations are not matching to one another. This could lead to an inhomogeneous motion path.
The aim of this approach is to add an extra weight to each node, which rewards more homogenous nodes. Hence, a closer area with a badly rated lower arm rotation is valued inferior to an area with a well-rated lower arm rotation.

D. Obstacles

The object to grasp as well as the body parts such as fingers and legs are included into the planner as obstacles. Since collision checks can be very expensive, we only test whether the fingers collide at this time. Furthermore, we only use cylinders (orange in Fig. 4). In doing so all obstacle positions except the fingers are calculated once per frame. In contrast finger positions are calculated for each node that is to be evaluated. These finger positions are represented by green lines in Fig. 4.
In order to check whether any part of the finger is also an obstacle part, the shortest distance or the intersection, respectively, between each obstacle and each finger is calculated. Thereby, if the shortest distance between one finger and an obstacle is smaller than the addition of obstacle radius and finger radius, a collision is detected. In that case, the corresponding node is rejected as possible way point of the path.

IV. Experiments

We tested the planning algorithm with different heuristic parameters as shown in Tab. I. All tests were made on a Nao robot using its 500 MHz X86 AMD GEODE processor with 256 MB SDRAM. The frame rate of the planner was 10 Hz. The first parameter set from Tab. I generates a heuristic that not only is fast with the planning algorithm used but also produces straight motion plans with homogenous lower arm rotations. This heuristic takes all heuristic equations from Sec. III-C into account. This means that nodes with more suitable lower arm rotations are rated superior to nodes with mismatching rotations. Not only is the angular deviation taken into account, but also the distance between the corresponding elbow positions. As a result, very few nodes need to be processed, which also can be seen in the right half of Fig. 5.
The second parameter set from Tab. I is in contradiction to the first one, because the heuristic does not consider the possible lower arm rotations. In comparison, the heuristic

TABLE I
Heuristic parameters in comparison to calculation time and nodes processed per frame.

p_0	p_1	p_2	p_3	p_4	$\sum nodes$	$\frac{ms}{frame}$
10	1	1	1	1	507	18 ms
1	0	0	0	0	703	35 ms
0	1	1	1	1	2566	$>$ 130 ms
0	0	0	0	0	68581	n/A

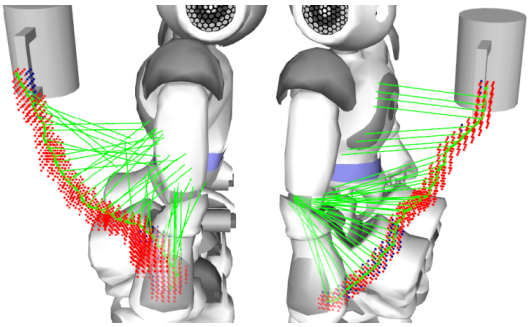


Fig. 5. Left: Depiction of nodes evaluated by using the second parameter set from Tab. I. Nodes considered are represented as red dots, nodes rejected due to possible obstacle collisions are represented as blue dots. The path planned and the corresponding lower arm rotations per way point are marked by the green lines. Right: Depiction of nodes evaluated by using first parameter set from Tab. I.

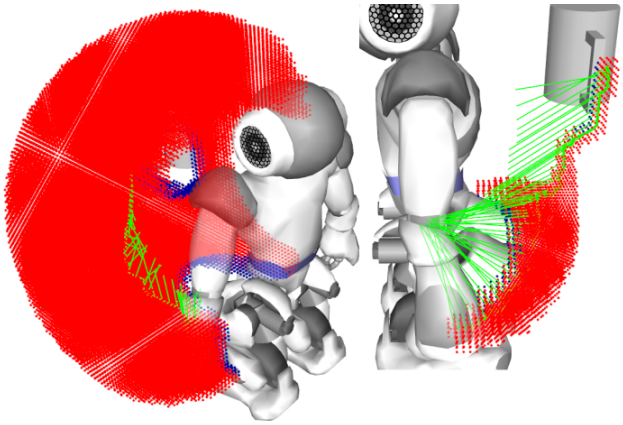


Fig. 6. Left: Depiction of nodes evaluated by using the last parameter set from Tab. I. Right: Depiction of nodes evaluated by using the third parameter set from Tab. I. The planning direction is from the object to the hand.

compels that many nodes need to be processed and consequently, the computation is distinctively slower. Since only nodes that are closer to the target node are rated superior, some elusive nodes are included into the path. Furthermore, on the left side in Fig. 5, the problem of inhomogeneous motions is conspicuous.

The right half of Fig. 6 constitutes the third parameter set from Tab. I tested. The corresponding heuristic only considers the connection of the lower arm rotations between the nodes. It does not evaluate the distance to the target node. Due to the lower arm target rotation, the heuristic compels that more nodes near the target node need to be processed. This parameter set is not suitable for real-time grasping, but generates very homogenous motion paths. Furthermore, during planning, a great many of nodes need to be processed. The last parameters set from Tab. I generates no heuristic at all. Since nearly every node of the reachability map needs to be processed, the computation time is very large and could not be measured on the Nao. In spite of that, we calculated a path using no heuristic in the simulation, which is pictured in the left half of Fig. 6.

V. Conclusion

Summarizing, the grasp function proposed is not only able to plan and execute a grasp on the affordable Nao robot, but it also works at 10 Hz. It seems that a predefined workspace can have very positive effects on the planning performance in such a way that a sufficient amount of possible way points can be processed. Furthermore, the search space can be decreased by including the evaluation of the lower arm rotation into the heuristic.

References

[1] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier, "Mechatronic design of nao humanoid," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, may 2009, pp. 769 –774.

[2] C. Borst, T. Wimbock, F. Schmidt, M. Fuchs, B. Brunner, F. Zacharias, P. R. Giordano, R. Konietschke, W. Sepp, S. Fuchs, C. Rink, A. Albu-Schaffer, and G. Hirzinger, "Rollin' justin - mobile platform with variable base," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, may 2009, pp. 1597 –1598.

[3] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura, "The intelligent asimo: system overview and integration," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 3, 2002, pp. 2478 – 2483 vol.3.

[4] T. Röfer, T. Laue, J. Müller, A. Fabisch, F. Feldpausch, K. Gillmann, C. Graf, T. J. de Haas, A. Härtl, A. Humann, D. Honsel, P. Kastner, T. Kastner, C. Könemann, B. Markowsky, O. J. L. Riemann, and F. Wenk, "B-human team report and code release 2011," 2011, only available online: http://www.b-human.de/downloads/bhuman11_coderelease.pdf.

[5] J. Forlizzi and C. DiSalvo, "Service robots in the domestic environment: a study of the roomba vacuum in the home," in *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, ser. HRI '06. New York, NY, USA: ACM, 2006, pp. 258–265.

[6] G. Cotugno and H. Mellmann, "Dynamic motion control: Adaptive bimanual grasping for a humanoid robot," in *Proceedings of the Workshop on Concurrency, Specification, and Programming CS&P 2010*, vol. Volume 2, Börnicke (near Berlin), Germany, September 2010.

[7] C. Borst, M. Fischer, and G. Hirzinger, "Grasping the dice by dicing the grasp," in *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 4, oct. 2003, pp. 3692 – 3697 vol.3.

[8] K. Lalibert´e, L. Birglen, and C. M. Gosselin, "Underactuation in robotic grasping hands," in *Journal of Machine Intelligence and Robotic Control*, vol. 4, no. 3, 2002, pp. 77 –87.

[9] G. Kragten, A. Kool, and J. Herder, "Ability to hold grasped objects by underactuated hands: Performance prediction and experiments," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, may 2009, pp. 2493 –2498.

[10] F. Zacharias, C. Borst, and G. Hirzinger, "Object-specific grasp maps for use in planning manipulation actions," in *Advances in Robotics Research*, T. Kröger and F. M. Wahl, Eds. Springer Berlin Heidelberg, 2009, pp. 203–213.

[11] ——, "Capturing robot workspace structure: representing robot capabilities," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 29 2007-nov. 2 2007, pp. 3229 –3236.

[12] E. Saff and A. Kuijlaars, "Distributing many points on a sphere," *The Mathematical Intelligencer*, vol. 19, pp. 5–11, 1997.

[13] J. Müller, T. Laue, and T. Röfer, "Kicking a ball - modeling complex dynamic motions for humanoid robots," in *RoboCup 2010: Robot Soccer World Cup XIV*, ser. Lecture Notes in Artificial Intelligence, J. R. del Solar, E. Chown, and P. G. Ploeger, Eds., vol. 6556. Springer; Heidelberg; http://www.springer.de/, 2011, pp. 109–120.

[14] T. D. DeRose and B. A. Barsky, "Geometric continuity, shape parameters, and geometric constructions for catmull-rom splines," *ACM Trans. Graph.*, vol. 7, no. 1, pp. 1–41, Jan. 1988.

Towards an industrial implementation of the walk-through programming technique for robotic manipulators

Luca Bascetta, Gianni Ferretti, Gianantonio Magnani, Paolo Rocco,
Fabio Abbà, Gian Paolo Gerio and Fabrizio Romanelli

***Abstract*— The present paper addresses some of the issues that should be covered in order to develop the walk-through programming technique in an industrial scenario. Besides presenting an exact formulation of the dynamics of the tool the teacher should feel when interacting with the robot, and a way to implement such dynamics on an industrial robot equipped with a wrist force/torque sensor, the paper addresses safety issues related to walk-through programming.**
An experiment on an industrial robot, involving both translational and rotational motion, is shown and discussed as well.

I. INTRODUCTION

Modern industrial robots are complex and powerful machines, able to execute a variety of different tasks with high speed and accuracy. Nevertheless, they still have a low degree of autonomy and adaptability, always needing huge efforts of a human operator to learn new tasks or tune existing ones.
A manual generation of a finishing path, for example, is a very complex and tedious task, taking up to ten weeks to create the path program to deburr a single aluminium wheel [1]. This is a rather common and crucial aspect, even for tasks which differ from finishing operations, that prevents robots from spreading in small and medium enterprises (SMEs), where mid/low lot size production does not allow for such a costly and time consuming operation. Generally, the fact that the programming phase is complex and time consuming is considered one of the major weaknesses of today's industrial robotic systems.

The introduction of an innovative programming paradigm, based on physical interaction between the human operator and the robot, represents a step forward in making the programming phase simple, intuitive and faster, and even in promoting an increased autonomy and cognitive ability of the robotic system.
In this new paradigm, the so-called walk-through programming, the human operator plays the role of a teacher that physically moves ('walks') the robot through the desired positions within the robot's working envelope. During this time, the robot's controller may scan and store coordinate values [2]. At the end of this operation the robot controller has recorded all the significant points in the trajectory demonstrated by the human, and is thus able to interpolate it and play it back.
Though it has been a step forward in programming techniques, the walk-through programming entails implementation and safety issues. Concerning the implementation, it must be noticed that the walk-through programming usually requires significant changes in the robot control software, that can be accomplished only with open robot control platforms. On the other side, safety is an important issue as the walk-through method requires the teacher to be within the robot's working envelope with the robot's controller energised: the person doing the teaching is thus in a potentially hazardous position [2]. Moreover, using an open robot control platform, that is not usually safety-rated as a commercial industrial controller, is a safety issue per se.

For the sake of completeness, it must be noticed that the walk-through programming is of particular interest in robotic surgery as well [3], [4], [5], [6]. In fact, it allows for a synergy between the surgeon and the robot: the robot provides geometric accuracy and increases safety preventing the execution of operations outside a predefined safe region; the surgeon guides the robot using his/her superior human senses and understanding of the overall situation to perform the surgery.

Walk-through programming is generally performed through force/torque sensors mounted on the robot wrist and is based on admittance control. Examples can be found in arm-manipulator coordination for load sharing [7], [8], [9], [10], and in industrial applications like welding [11] and spray painting [12].
The high cost of force/torque sensors, however, fosters the development of alternative devices, as the adoption of such a sensor can be considered sensible only when the industrial application, that will be teached using the walk-through programming, already requires a force/torque measurement. An example of walk-through programming based on a different device, namely a 6-d.o.f. mouse, was presented in [13].

The present paper extends the work presented in [12], proposing an implementation of the walk-through programming technique that takes into account all the characteristics of an industrial scenario, being compliant with the industrial safety standards, and that allows changes in the tool position and orientation as well, as shown in the experiment here reported.

The research leading to these results has received funding from the European Community's **Seventh Framework Programme** FP7/2007-2013 - Challenge 2 - **Cognitive Systems, Interaction, Robotics** - under grant agreement No 231143 - **ECHORD** (Experiment **FIDELIO**).

L. Bascetta, G. Ferretti, G. Magnani and P. Rocco are with Politecnico di Milano, Dipartimento di Elettronica e Informazione, Piazza Leonardo Da Vinci 32, 20133, Milano, Italy (email: {bascetta, ferretti, magnani, rocco}@elet.polimi.it).

F. Abbà, G.P. Gerio and F. Romanelli are with Comau Robotics, Via Rivalta 30, 10095, Grugliasco (Torino), Italy.

II. DYNAMICS OF THE VIRTUAL TOOL

Considering an industrial application, like metal finishing or painting operations: the physical interaction between the human operator and the robot should be conceived in such a way that the teacher has the impression to grab a real tool, e.g. a deburring tool or a spray gun, instead of the robot end-effector. The role of the control system is thus to accommodate for the motion commanded by the teacher, mimicking the same dynamic behaviour of the real tool, i.e. behaving like a virtual tool that exhibits the same mechanical properties of the real tool.
The first step towards the development of the walk-through programming technique, is thus the derivation of the exact Newton-Euler dynamic equations of a virtual tool.

Consider Fig. 1, where the following frames are reported: (x_0, y_0, z_0), the absolute frame; (x_T, y_T, z_T), a frame attached to the robot end-effector; (x_S, y_S, z_S), a frame attached to the force/torque sensor; (x_P, y_P, z_P), a frame attached to the actual tool, with the origin located where the operator grabs the tool; (x_B, y_B, z_B), a frame attached to the virtual tool, with the origin located in its center of mass.

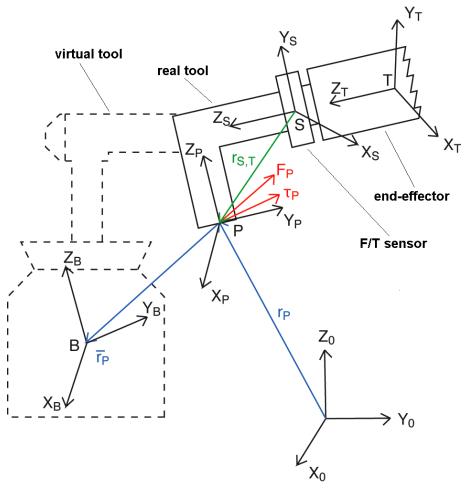


Fig. 1. The frames used to formulate the dynamics of the virtual tool.

The dynamic model is written using the following coordinates

$$ {}^0\mathbf{r}_P = \begin{bmatrix} x_P \\ y_P \\ z_P \end{bmatrix} \quad \text{and} \quad \mathbf{p}_P = \begin{bmatrix} e_{P,0} \\ e_{P,1} \\ e_{P,2} \\ e_{P,3} \end{bmatrix} \tag{1} $$

which represent the coordinates of the origin of the frame P with respect to the reference frame 0 and the relative orientation of the same frames, respectively. Notice that the orientation in (1) is expressed in terms of the four Euler parameters (unit quaternion): although being redundant, this representation avoids the so-called *gimbal lock*[1].

[1]With the expression *gimbal lock* we mean the situation when two axes are aligned, and the system looses one rotational d.o.f.

The equations of motion of the virtual tool based on Newton-Euler dynamics can thus be written as

$$ {}^0\mathbf{f}_P = m\left[{}^0\mathbf{a}_P - {}^0\mathbf{g} + {}^0\boldsymbol{\varepsilon}_P \times {}^0\mathbf{r}_{B,P} + {}^0\boldsymbol{\omega}_P \times \left({}^0\boldsymbol{\omega}_P \times {}^0\mathbf{r}_{B,P}\right)\right] + \mathbf{D}'\,{}^0\mathbf{v}_P \tag{2} $$

$$ {}^0\boldsymbol{\tau}_P = {}^0\mathbf{I}_B\,{}^0\boldsymbol{\varepsilon}_P + {}^0\boldsymbol{\omega}_P \times \left({}^0\mathbf{I}_B\,{}^0\boldsymbol{\omega}_P\right) + {}^0\mathbf{r}_{B,P} \times {}^0\mathbf{f}_P + \mathbf{D}''\,{}^0\boldsymbol{\omega}_P \tag{3} $$

where the following symbols have been used:

- m, mass of the virtual tool;
- ${}^0\mathbf{I}_B$, inertia tensor of the virtual tool with respect to a frame B, with origin in the center of mass of the virtual tool and with the same orientation of frame P, expressed in the absolute frame;
- ${}^0\mathbf{g}$, gravity acceleration vector, expressed in the absolute frame;
- ${}^0\mathbf{f}_P$, ${}^0\boldsymbol{\tau}_P$, vector of forces and moments applied by the human operator to the virtual tool, expressed in the absolute frame;
- ${}^0\mathbf{r}_{B,P}$, position of the center of mass of the virtual tool with respect to the origin of the frame P, expressed in the absolute frame;
- $\mathbf{D}'$, $\mathbf{D}''$, viscous friction matrix for linear and rotational motions of the virtual tool.

In equations (2)-(3) the following kinematic terms have been introduced as well:

- the absolute velocity of the origin of the frame P, expressed in frame 0

$$ {}^0\mathbf{v}_P = \frac{d^0\mathbf{r}_P}{dt} = \begin{bmatrix} v_{P,x} \\ v_{P,y} \\ v_{P,z} \end{bmatrix} $$

- the absolute acceleration of the origin of the frame P, expressed in frame 0

$$ {}^0\mathbf{a}_P = \frac{d^0\mathbf{v}_P}{dt} = \begin{bmatrix} a_{P,x} \\ a_{P,y} \\ a_{P,z} \end{bmatrix} $$

- the angular velocity of the frame P, expressed in frame 0

$$ {}^0\boldsymbol{\omega}_P = 2\begin{bmatrix} -e_{P,1} & e_{P,0} & -e_{P,3} & e_{P,2} \\ -e_{P,2} & e_{P,3} & e_{P,0} & -e_{P,1} \\ -e_{P,3} & -e_{P,2} & e_{P,1} & e_{P,0} \end{bmatrix} \frac{d\mathbf{p}_P}{dt} = 2\mathbf{E}\frac{d\mathbf{p}_P}{dt} $$

- the angular acceleration of the frame P, expressed in frame 0

$$ {}^0\boldsymbol{\varepsilon}_P = 2\frac{d\mathbf{E}}{dt}\frac{d\mathbf{p}_P}{dt} + 2\mathbf{E}\frac{d^2\mathbf{p}_P}{dt^2} $$

- the rotation matrix of the frame P with respect to the frame 0

$$ {}^0\mathbf{R}_P = 2\begin{bmatrix} {}^0\mathbf{n}_{x,P} & {}^0\mathbf{n}_{y,P} & {}^0\mathbf{n}_{z,P} \end{bmatrix} $$

where

$$ {}^0\mathbf{n}_{x,P} = \begin{bmatrix} e_{P,0}^2 + e_{P,1}^2 - \frac{1}{2} \\ e_{P,1}e_{P,2} + e_{P,0}e_{P,3} \\ e_{P,1}e_{P,3} - e_{P,0}e_{P,2} \end{bmatrix} $$

$$ {}^0\mathbf{n}_{y,P} = \begin{bmatrix} e_{P,1}e_{P,2} - e_{P,0}e_{P,3} \\ e_{P,0}^2 + e_{P,2}^2 - \frac{1}{2} \\ e_{P,2}e_{P,3} + e_{P,0}e_{P,1} \end{bmatrix} $$

$$ {}^0\mathbf{n}_{z,P} = \begin{bmatrix} e_{P,1}e_{P,3} + e_{P,0}e_{P,2} \\ e_{P,2}e_{P,3} - e_{P,0}e_{P,1} \\ e_{P,0}^2 + e_{P,3}^2 - \frac{1}{2} \end{bmatrix} $$

and

$$ e_{P,0}^2 + e_{P,1}^2 + e_{P,2}^2 + e_{P,3}^2 = 1 $$

Note that in equations (2)-(3) the inertia tensor of the virtual tool, ${}^0\mathbf{I}_B$, is time varying, as it is expressed with reference to the absolute frame. Alternatively, the equations can be rewritten by projecting all the quantities onto the frame P attached to the real, and thus also to the virtual, tool.
The dynamic equations (2)-(3) can thus be rewritten as

$$ {}^P\mathbf{f}_P = m\left[{}^P\mathbf{a}_P - {}^P\mathbf{g} + {}^P\varepsilon_P \times {}^P\mathbf{r}_{B,P} + {}^P\omega_P \times \left({}^P\omega_P \times {}^P\mathbf{r}_{B,P}\right)\right] + \mathbf{D}' \, {}^P\mathbf{v}_P \quad (4) $$

$$ {}^P\tau_P = {}^P\mathbf{I}_B \, {}^P\varepsilon_P + {}^P\omega_P \times {}^P\mathbf{I}_B \, {}^P\omega_P + {}^P\mathbf{r}_{B,P} \times {}^P\mathbf{f}_P + \mathbf{D}'' \, {}^P\omega_P \quad (5) $$

In this formulation the inertia tensor ${}^P\mathbf{I}_B$ referred to the frame P is used, which is now time invariant.

Notice that the frame used to command the robot motion is the sensor frame S. Specifically the motion will be assigned commanding the motion of the origin of the frame S with respect to the frame 0 and the rotation matrix of the said frame. The following parameters of the real tool are introduced:

- ${}^S\mathbf{r}_{S,P}$, position of the origin of the frame P with respect to the origin of the frame S, expressed in the frame S;
- ${}^S\mathbf{R}_P$, rotation matrix of the frame P with respect to the sensor frame S.

The position and orientation of the frame S are thus readily given by

$$ {}^0\mathbf{r}_S = {}^0\mathbf{r}_P - {}^0\mathbf{R}_P \left({}^S\mathbf{R}_P\right)^T {}^S\mathbf{r}_{S,P} \qquad {}^0\mathbf{R}_S = {}^0\mathbf{R}_P \left({}^S\mathbf{R}_P\right)^T $$

Finally, the relation between the forces and moments measured by the sensor in the frame S and those applied at the frame P, under the hypothesis that the mass of the virtual tool is negligible, are given by

$$ {}^P\mathbf{f}_P = \left({}^S\mathbf{R}_P\right)^T {}^S\mathbf{f}_S \qquad {}^P\tau_P = \left({}^S\mathbf{R}_P\right)^T \left({}^S\tau_S - {}^S\mathbf{r}_{S,P} \times {}^P\mathbf{f}_P\right) $$

III. AN ADMITTANCE CONTROLLER

In order to ensure that the robot behaves as the virtual tool, in response to the teacher's forces, a controller that enforces the dynamics described by equations (4)-(5) must be devised. It must be noticed, however, that (4)-(5) are a set of six nonlinear and coupled differential algebraic equations whose integration in the real-time robot controller is indeed a challenging task. Furthermore, considering that the teaching phase is characterised by low linear, and even lower, angular velocities and accelerations, the couplings between the Newton and Euler dynamics can be neglected.

The dynamic equations in (4)-(5) can be thus simplified as follows

$$ {}^P\mathbf{f}_P + m^P\mathbf{g} = m^P\mathbf{a}_P + \mathbf{D}' \, {}^P\mathbf{v}_P $$
$$ {}^P\tau_P + {}^P\mathbf{f}_P \times {}^P\mathbf{r}_{B,P} = {}^P\mathbf{I}_B \, {}^P\varepsilon_P + \mathbf{D}'' \, {}^P\omega_P $$

In view of these simplified relations, the dynamics of the virtual tool can be enforced by way of an admittance controller: such a controller accepts forces and moments and yields corresponding displacements.

IV. SAFETY ASPECTS IN WALK-THROUGH PROGRAMMING

Industrial robots are designed to work in a highly structured environment, doing fast and accurate movements in an area that is rigidly separated, by way of safety fences, from the areas occupied by humans. Robots safety requirements for industrial environments are regulated by the international standard ISO 10218-1 [14].
As already pointed out, walk-through programming entails important safety issues, as the teacher is within the robot's working envelope with the robot's controller energised. In particular, the safety aspects that have to be considered concern the ergonomic design and the safe motion of the robot during human-robot interaction.

Considering a single human operator interacting with a single robot, the following situations occur:

- Activation of drive power
  Drive power has to be activated keeping outside the robot area, after checking that nobody is in the robot area (as described by general safety rules).
  Once the drive power has been activated, the teacher can start the walk-through programming.

- Walk-through programming
  The teacher should grab and activate a three position enabling device, compliant with the requirements of 5.8.3 in [14], interlocked with the safety devices of the cell/robot control: as long as he/she holds the device in a centre-enabled position, robot motion is allowed.
  The robot motion is commanded by the teacher through the Teach Pendant or the device adopted for the walk-through programming.

According to [14] the speed of the end-effector mounting flange and of the tool centre point shall not exceed $250\,mm/s$, and the device adopted for the walk-through programming shall be located close to the end-effector.

V. WALK-THROUGH PROGRAMMING ON THE C4G OPEN

The experimental setup consists of a robot Smart Six (Fig. 2), a 6 d.o.f., 6 Kg payload industrial manipulator manufactured by COMAU, equipped with the open version of the COMAU C4G controller [15]. In the open version, the C4G, acting as a network client, is linked to a real-time external PC through a real-time ethernet connection based on the RTnet protocol [16]. The real-time external PC, acting as a network server, is based on the RTAI Linux real-time

extension [17]. The ATI 6-axes wrist force/torque sensor is fitted to the arm end-effector and linked to the PC through a DAQ board that is managed by the RTAI system, thanks to a real-time extension of the Comedi drivers [18].

Fig. 2. The Smart Six robot during a walk-through programming operation.

Considering the layout of the experiment, its aim, and the results of a risk assessment procedure, it turns out that three main actors participate to the walk-through programming task:

- the teacher, who physically walks the robot through the desired positions. According to [14], he/she holds the enabling device in a centre-enabled position, in order to allow for the activation of the drive power.
- the responsible of the Human Machine Interface, an expert that helps the teacher, who is probably not a robotics expert, dealing with the robot, and supervises the teaching process. Using the controller Teach Pendant he/she activates the drive power and gives consent to the external PC to send position set-points to the industrial controller.
- the robotic system expert, who sets up the real-time communication between the external PC and the open controller. He/she supervises the whole hardware-software setup of the experiment.

Notice that, in a future industrial implementation of the walk-through programming, all the required devices will be provided by the industrial controller. The teacher will program the robot through the Human Machine Interface, without the need of further operators.

The easiest way to realise the walk-through programming technique on an industrial robot controller is by enforcing the dynamics of the virtual tool through an admittance controller, as described in Section III. Furthermore, to implement such a controller, an external loop is closed outside the position control loops in the industrial controller: the admittance controller, fed by the measurements of forces and moments acquired by a force/torque sensor mounted on the end-effector, yields modifications to the motion control set-points

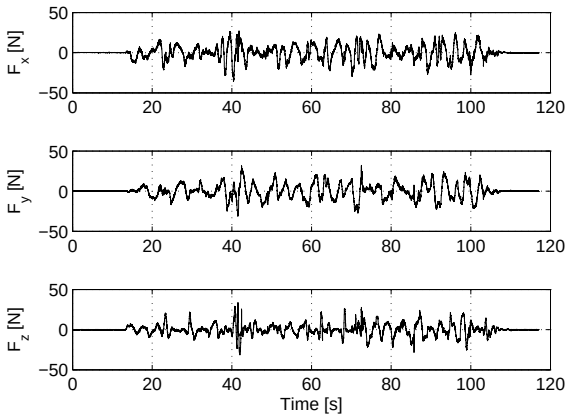


Fig. 3. Forces, expressed in the absolute frame, during a walk-through programming experiment.

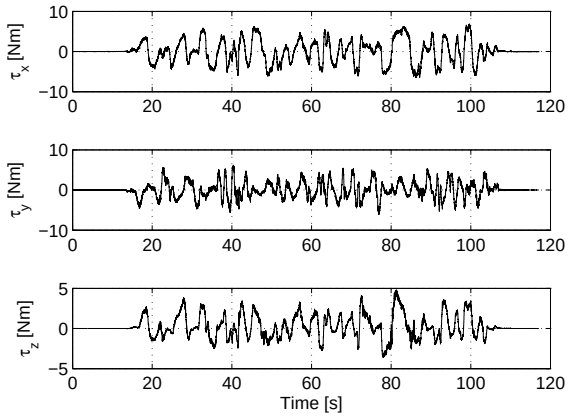


Fig. 4. Torques, expressed in the absolute frame, during a walk-through programming experiment.

in order to guarantee the prescribed compliant behaviour. Furthermore, in order to fulfil all the safety requirements discussed in Section IV, a set of safety constraints is introduced: a virtual force proportional to the tool centre point Cartesian velocity, by way of a nonlinear viscous friction coefficient that rapidly increases as the velocity approaches the value of $250\,mm/s$, is superimposed to the forces measured by the force/torque sensor. This force prevents the human operator from exceeding the maximum velocity allowed by the safety requirements[2].

An experiment has been conducted where the teacher walks the end-effector around a worktable. The virtual tool was given the mechanical properties of a sphere of $10\,Kg$ of mass and $5\,cm$ of radius, with a constant viscous friction coefficient of $50\,Kgm/s$ along each translational direction, and $10\,Kgrad/s$ along each rotational direction.

Figs. 3, 4, 5 and 6 show the forces and moments (expressed in the absolute frame) and the corresponding virtual tool

[2]In case of failure of these safety constraints, a further check on the control software stops the robot.

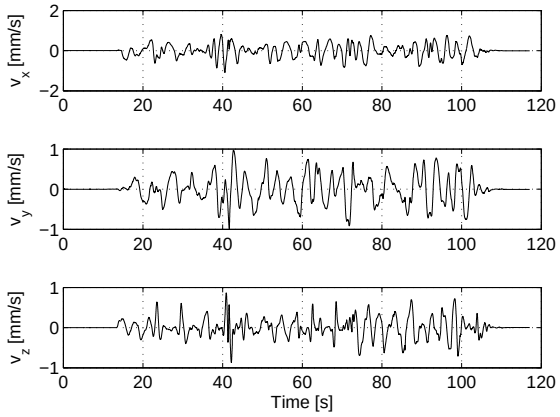


Fig. 5. Virtual tool linear velocity, expressed in the absolute frame, during a walk-through programming experiment.

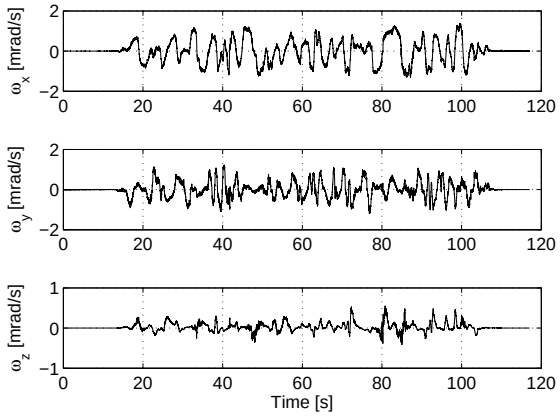


Fig. 6. Virtual tool angular velocity, expressed in the absolute frame, during a walk-through programming experiment.

linear and angular velocities (expressed in the absolute frame).

Notice that the walk-through programming experiment here depicted is characterised by several movements, involving tool position and orientation. Furthermore, though the accelerations exerted by the teacher are sometimes even larger than the ones applied by an actual human operator during a teaching operation, the robot always smoothly accommodates for the teacher's commands.

VI. CONCLUSIONS

Starting from a previous work, this paper addresses some of the issues that should be covered in order to fill the gap between an academic proof-of-concept and an industrial implementation of the walk-through programming.

An exact formulation of the dynamic equations of a virtual tool, together with the use of an impedance controller based on force/torque measurements, allow to achieve an easy and smooth interaction between the teacher and the industrial robot. The results of a walk-through programming experiment show the behaviour of the system in the presence of several different movements, involving changes in the tool position and orientation.

REFERENCES

[1] C. Powell, "Case study - kuntz electroplating automated polishing system," Robotics Online (www.robotics.org), 2002.

[2] A. K. Gupta and S. K. Arora, *Industrial automation and robotics*. Laxmi Publications LTD, 2007.

[3] "ECHORD experiment HipROB – Robot-assisted and ultrasound-guided navigation for hip resurfacing arthroplasty," http://www.echord.info/wikis/website/hiprob.

[4] T. Ortmaier, H. Weiss, U. Hagn, M. Grebenstein, M. Nickl, A. Albu-Schäffer, C. Ott, S. Jörg, R. Konietschke, L. Le-Tien, and G. Hirzinger, "A hands-on-robot for accurate placement of pedicle screws," in *IEEE International Conference on Robotics and Automation*, May 2006, pp. 4179–4186.

[5] M. Jakopec, S. J. Harris, F. Rodriguez y Baena, P. Gomes, J. Cobb, and B. L. Davies, "The first clinical application of a "hands-on" robotic knee surgery system," *Computer Aided Surgery*, vol. 6, no. 6, pp. 329–339, 2001.

[6] R. Kumar, P. Berkelman, P. Gupta, A. Barnes, P. S. Jensen, L. L. Whitcomb, and R. H. Taylor, "Preliminary experiments in cooperative human/robot force control for robot assisted microsurgical manipulation," in *IEEE International Conference on Robotics and Automation*, April 2000, pp. 610–617.

[7] O. M. Al-Jarrah and Y. F. Zheng, "Arm-manipulator coordination for load sharing using compliant control," in *IEEE International Conference on Robotics and Automation*, April 1996, pp. 1000–1005.

[8] ——, "Arm-manipulator coordination for load sharing using variable compliance control," in *IEEE International Conference on Robotics and Automation*, April 1997, pp. 895–900.

[9] ——, "Arm-manipulator coordination for load sharing using reflexive motion control," in *IEEE International Conference on Robotics and Automation*, April 1997, pp. 2326–2331.

[10] T. Tsumugiwa, R. Yokogawa, and K. Yoshida, "Stability analysis for impedance control of robot for human-robot cooperative task system," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2004, pp. 3883–3888.

[11] M. H. Ang Jr and L. S. Yong, "An industrial application of control of dynamic behavior of robots – A walk-through programmed welding robot," in *IEEE International Conference on Robotics and Automation*, April 2000, pp. 2352–2357.

[12] G. Ferretti, G. Magnani, and P. Rocco, "Assigning virtual tool dynamics to an industrial robot through an admittance controller," in *IEEE International Conference on Advanced Robotics*, June 2009, pp. 1–6.

[13] G. P. Gerio, L. Lacchello, F. Romanelli, B. Cicciarello, M. Fraccaroli, L. Molinari Tosatti, D. Parazzoli, E. Scari, and M. Danesi, "EP2194434 (A1) – Manual Guidance device for robot applications."

[14] *ISO 10218-1:2006 Robots for industrial environments – Safety requirements – Part 1: Robot*, 2006.

[15] *Comau Robotics instruction handbook – C4G OPEN System Software Rel. 3.33*, 2011.

[16] "RTnet – Hard real-time networking for real-time Linux," http://www.rtnet.org.

[17] "RTAI – Real Time Application Interface," http://www.rtai.org.

[18] "COMEDI – Linux COntrol and MEasurement Device Interface," http://www.comedi.org.

Generation of optimal trajectories for industrial robots - a test case manipulating a glass of water at high velocities

Martin L. Felis, Benjamin Reh, Katja Mombaur
Interdisciplinary Center for
Scientific Computing (IWR)
University of Heidelberg, INF 368
D - 69120 Heidelberg, Germany
martin.felis@iwr.uni-heidelberg.de

***Abstract*— In this paper we use optimal control methods to generate robot motions that are subject to acceleration constraints on the end effector. This approach allows us to compute motions that are both feasible for the robot and feature complex properties for the whole trajectory. We modeled the dynamics of the industrial robot arm, namely the KR5 SIXX R850 using a rigid-body model. We formulated an optimal control problem that allows us to compute optimal trajectories between given configurations that minimizes the acceleration of the center of a glass of water. We computed optimal trajectories for two scenarios and successfully transferred on the real robot which was able to move a glass of water without spilling water.**

I. INTRODUCTION

A. The ECHORD experiment GOP - Generating optimal paths

The generation of the best possible path that does not violate any constraints imposed by the environment is an ubiquitous task in both industrial and humanoid robotics. Currently there is no algorithmic approach available that allows to address this problem for very complex dynamic robot systems in cluttered changing environments in real time. Instead there are two established but still quite separated research areas that both address a part of the problem, namely path planning and numerical optimal control. Path planning is mainly interested in the determination of a feasible path in very complex environments based on geometric and kinematic models [1]. Numerical optimal control techniques [6], [7] are capable to generate optimal trajectories for robot manipulators or humanoid robots taking into account the dynamics [2]; however the treatment of a large number of environmental constraints giving rise to many local minima makes the problems very hard, if not impossible, to solve.

This project aims at combining state of the art developments of these two domains and to create the algorithmic foundations to tackle real time optimal control problems in cluttered environments. It builds on top of the experiences of the two partner institutions IWR at the University of Heidelberg and LAAS-CNRS in Toulouse - in the respective areas. When this issue is solved it will be beneficial to nearly all the defined scenarios and research foci of the ECHORD project. More specifically, in the Hyper-Flexible Cells scenario, it is crucial to determine if a robot will be capable to handle the multitude of tasks it will be asked to do and to optimize the robots motions for the different steps, tools and manipulation of different work pieces. A changing workflow and the interaction with other robots and humans in the first and third scenario emphasize the need for algorithms to efficiently re-generate the best possible motions under varying circumstances.

Fig. 1. KR5 SIXX R850 with gripper and water glass.

In the ECHORD GOP project, two robot platforms are used, namely the humanoid robot HRP-2 available at LAAS, as well as a small industrial robot arm KUKA KR5 SIXX R850 at IWR. The latter is used for the test case demonstration in this paper.

B. A test case imposing acceleration constraints: manipulating a glass of water at high velocities

In this paper, we treat a first test case for the generation of optimal paths in industrial robot arms. It is not an industrial application setting, but a small example which however is characterized by complex constraints. The task is to move a glass filled with water over some distance and at fast speed, starting and stopping at rest. The goal obviously is to not spill any water during the maneuver, which is particularly challenging in the initial and final phase with large accelerations an decelerations. The KUKA robot KR5 SIXX R850 is equipped with a SCHUNK gripper and a customized glass holder created on a RepRap 3D printer (see Fig. 1).

Presently, the arm is still moving in free space and only has to avoid self collision, but complex path constraints of the environment will be added in the course of this project. For the present example, only efficient optimal control techniques are required, but for the follow-up a combined approach of optimal control and path planning techniques will be applied which is currently developed in the ECHORD-GOP project together with our partners at LAAS.

II. DYNAMIC MODELING OF THE ROBOT ARM

The KUKA robot arm has 6 axes and a length of 850mm when fully stretched and weighs 29kg.

Using generalized positions $q(t)$ and velocities $\dot{q}(t)$, the dynamics of the robot can be expressed as:

$$M(q)\ddot{q} = -N(q,\dot{q}) + \tau, \tag{1}$$

where $M(q)$ is the joint space inertia matrix, $N(q,\dot{q})$ are the coriolis and gravitational forces, and τ are the torques applied at the joints.

We also model the effects of the motor inertias $\mu_1,\ldots,\mu_6$ of the actuators into account. This is done by formulating a new inertia matrix $\tilde{M}(q)$ as:

$$\tilde{M}(q) = M(q) + diag(\mu_1,\ldots,\mu_6) \tag{2}$$

This matrix is then used in Eq. 1.

The dynamic model for the robot was created using original dynamic parameters such as the locations for the centers of mass and the inertia of the individual segments. For the modeling we used the RBDL [4] which also performs the computations of the dynamics. It uses the Composite Rigid Body Algorithm for the computation of $M(q)$ and the Recursive Newton Euler Algorithm to compute $N(q,\dot{q})$. The library is implemented using spatial algebra for a clear yet efficient representation of the algorithms [5].

III. OPTIMAL CONTROL OF ROBOT MOTIONS

A significant advantage of optimization over approaches based on pure simulation is that simulation always requires to fix important quantities in advance: if it is performed on a forward dynamics model the input forces and torques have to be pre-specified to obtain the resulting motion, and if performed on an inverse dynamics model the position and velocity histories have to be fixed to be able to calculate the required driving torques and forces. However, typically none of the quantities is exactly known a priori. Optimization-based simulation - or optimal control - allows to leave forces, torques $u(t)$ and the motion $x(t) = [q(t),\dot{q}(t)]^T$, free and to determine all these quantities simultaneously according to some desired optimization criterion.

The optimal control problem can be written as:

$$\min_{x(\cdot),u(\cdot),t_f} \quad \int_0^{t_f} \phi(x(t),u(t))dt \tag{3}$$

subject to:

$$\begin{aligned} \dot{x}(t) &= f(t,x(t),u(t)) && (4)\\ g(t,x(t),u(t),p) &\geq 0, && (5)\\ r(x(0),x(t_f),p) &= 0. && (6) \end{aligned}$$

The objective function (3) is of the form:

$$\phi(x(t),u(t)) = a_x(t)^2 + a_y(t)^2 + 0.001 * a_z(t)^2$$

where $a_x(t)$, $a_y(t)$, and $a_z(t)$ is the acceleration of the center point of the glass along its local x–, y–, and z–direction, where the z–axis describes the longitudinal axis of the glass. The terms for the accelerations along the local x– and y–axes make sure that the acceleration of the water towards the border of the glass is minimized. The additional term for the acceleration along the z–axis result in a smoothing of the overall acceleration profile.

The dynamics of the robot is described by (4). General state and control boundaries such as joint and torque limits are described by (5). Start and end configuration is modeled by the boundary conditions (6). These constraints at t_0 and t_f also require, that the acceleration of the center point of the glass is equal to the acceleration due to gravity to ensure valid start and end points. Furthermore the generalized velocity has to be zero at start and end point, i.e. $\dot{q}(t_0) = \dot{q}(t_f) = 0$. In addition, boundaries on the vertical accelerations are applied to make sure it always points downwards with respect to the glass.

To solve this problem the software package MUSCOD-II discretizes the continuous formulation (3)–(6) for both controls and states. The resulting nonlinear optimization problem is then solved by using a specially tailored sequential quadratic programming (SQP) method [6].

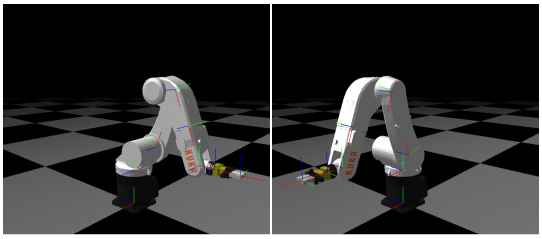
IV. EXPERIMENT

The firmware installed on the control unit limits the movement to a set of standard forms and does not allow a custom trajectory to be played. We use the Remote Sensor Interface (RSI) plugin provided by KUKA to overcome this problem.

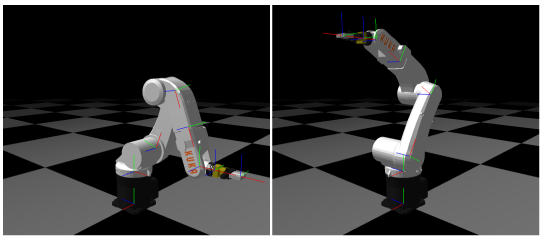
The RSI interface was originally designed to allow adjustments during the robot's movement with the help of external sensors. We use this interface to control the robot directly from a separate PC. The control unit established a TCP connection over Ethernet to the controlling computer. Both exchange information in a fixed time interval of 20ms which is also the interpolation time of the robot. The computer receives the current positions and sends the target positions of all 6 degrees of freedom.

The experiments show that there is a constant delay of $\approx$200ms between the provided angles and the actual angles of the joints. When neglecting this delay the calculated trajectory is very accurately reproduced on the robot (mean error $< 1^\circ$ on each joint).

Fig. 2 shows the two of the various scenarios for which we computed optimal trajectories. In scenario "rotation" (Fig. 2(a)) start and end conditions for the robot only differ in the 90° rotation of the first joint. In scenario "rotation and vertical translation" (Fig. 2(b)) the end state is the same as previously but with an additional translation of the glass along the vertical axis.

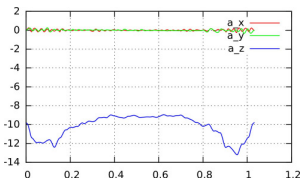


(a) Scenario: rotation

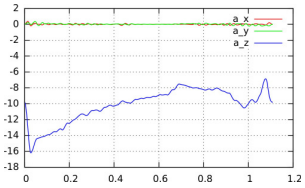


(b) Scenario: rotation and vertical translation

Fig. 2. Start and end positions for the two experiment setups for which we optimized the intermediate trajectories.



(a) Scenario: rotation



(b) Scenario: rotation and vertical translation

Fig. 3. Profile of the accelerations of the glass center in local coordinates for the optimal trajectories in both setups. The local z–axis is the longitudinal axis of the glass.

V. RESULTS AND OUTLOOK

Figures 4 and 5 shows the computed motions in the simulation and the actual robot performing the two maneuvers moving a glass of water. The duration of motion "rotation" only takes 1.03 seconds and the motion "rotation and vertical translation" has a duration of 1.11 seconds. An intuitive leaning of the glass can be seen that ensures acceleration along the vertical axis of the glass. In both cases, the optimized motions lead to splash-free maneuvers in reality.

A profile of the acceleration at the glass center in local coordinates can be found in Fig. 3. It can be seen that the vertical component (with respect to the glass) a_z stays negative while the orthogonal components a_x and a_y become very small due to the optimization criterion.

This work will be extended to even faster and more difficult maneuvers. We intend to produce motions with short phases where the glass is fully inverted, and will evaluate if the robots properties allow to perform such a challenging maneuver. We will also extend this work to scenarios where the robot's motion is constrained by complex obstacles. Finally, over the course of the GOP project we are looking a other test cases with very complex constraints and different local minimum trajectories where a combination of path planning and optimal control techniques is required.

REFERENCES

[1] S. M. LaValle and J. J. Kuffner: Rapidly-exploring random trees: Progress and prospects, in B. R. Donald, K. M. Lynch, and D. Rus, editors, Algorithmic and Computational Robotics: New Directions, pages 293–308. A K Peters, Wellesley, MA, 2001.

[2] G. Schultz, K. Mombaur: Modeling and Optimal Control of Human-Like Running, IEEE/ASME Transactions on Mechatronics, 2010, published online 2009.

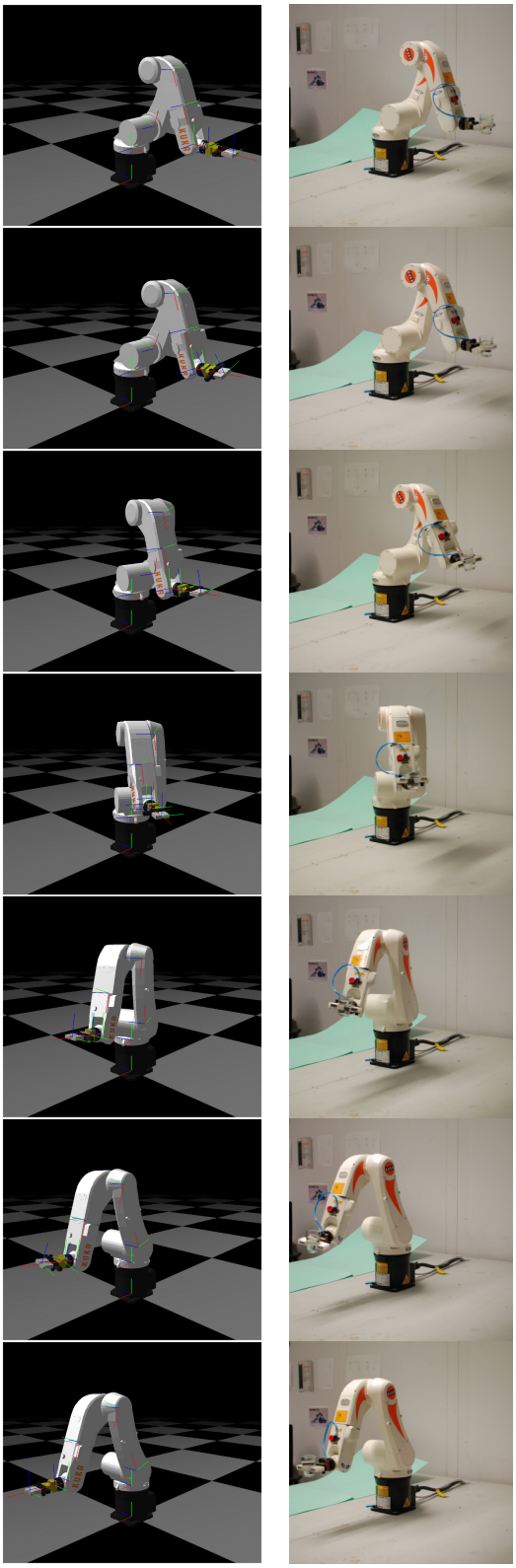
[3] K. Mombaur, Using optimization to create self-stable human-like running, Robotica, 2009, published online June 2008.

[4] M. L. Felis: RBDL – The Rigid Body Dynamics Library, http://rbdl.bitbucket.org, 2012.

[5] R. Featherstone: Rigid Body Dynamics Algorithms, Springer, New York, 2008.

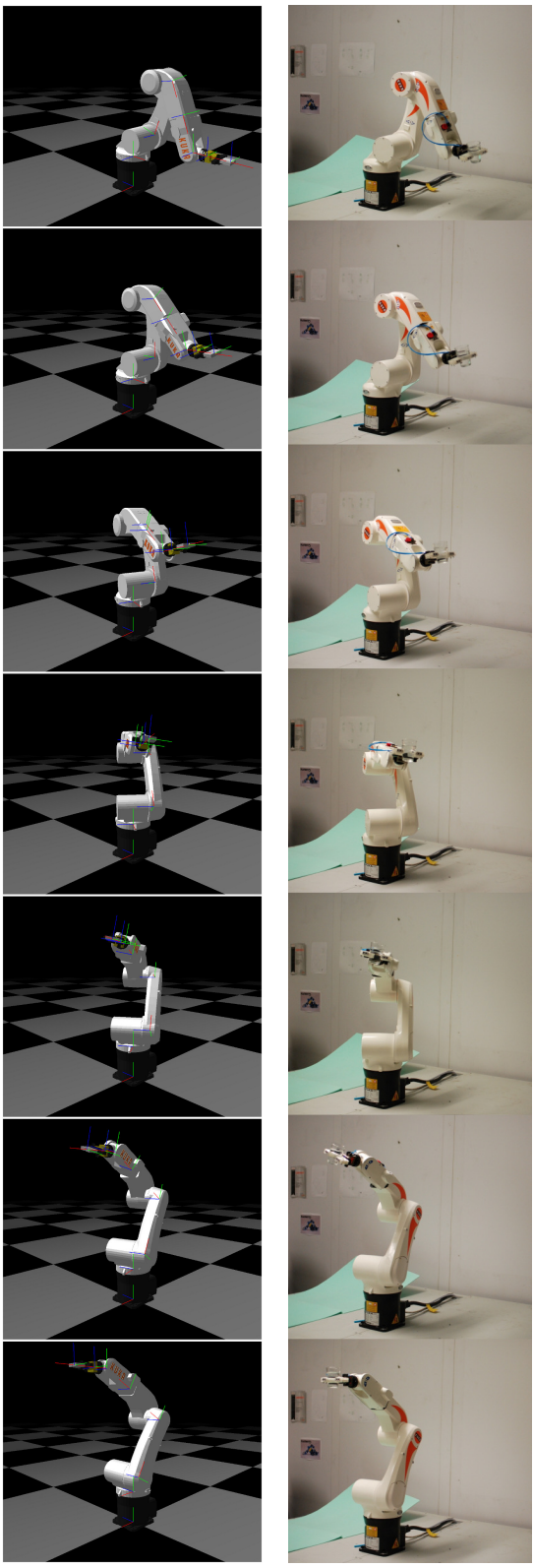
[6] D. B. Leineweber, I. Bauer, A. A. S. Schäfer, H. G. Bock and J. P. Schlöder, An Efficient Multiple Shooting Based Reduced SQP Strategy for Large-Scale Dynamic Process Optimization (Parts I and II), Computers and Chemical Engineering, 2003.

[7] H. G. Bock, K. Plitt, A multiple shooting algorithm for direct solution of optimal control problems, in Proceedings of the 9th IFAC World Congress Budapest, Pergamon Press, 1984.



(a) Simulation (b) Movement of the real robot

Fig. 4. Comparison of the simulated and actual robot motion for scenarion “rotation”.



(a) Simulation (b) Movement of the real robot

Fig. 5. Comparison of the simulated and actual robot motion for scenarion “rotation and vertical translation”.

A new cable-driven robotic arm for flexible and mobile applications

Roland Behrens, Andreas Hoffmann, Maik Poggendorf, Norbert Elkmann, Fraunhofer IFF (Germany)

***Abstract*— In this paper, a lightweight robotic arm based on igus® robolink is presented. First, a short introduction about the robot and its relevance is given. Next, the optimization of the workspace is determined. Then, a mechanical rope tensioner and gravity compensation as well as a novel drive system for compliant gripper is presented. Finally, a path planning algorithm and an independent position controller for the cable-driven robot is investigated. The robot was built-up in the ALEXA experiment of the ECHORD project.**

I. Introduction

The robotic arm presented here was developed in the ALEXA experiment. It has the capabilities of a mobile, lightweight manipulator based on the *robolink* construction kit from the company *igus*. The innovative system allows for lightweight manipulators to be easily built up for use in smart robotic applications. It consists of cable-driven joints and links of selectable lengths and materials. This option allows for a manipulator to be built for specific need. Similar works are presented in [1] and [2] amongst others.

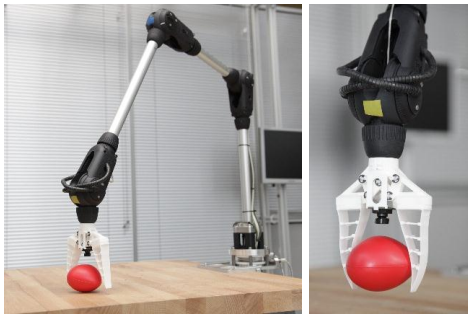


Figure 1. ALEXA robotic arm (left), compliant gripper (right)

Currently, *igus* does not offer a complete robotic arm that is ready to be used as a common tool for standard pick-and-place applications. In the ALEXA experiment, a drive and control system is integrated with a *robolink* robotic arm. Furthermore, the drive and control system can be fully detached from the robotic arm. This topology allows easy transportation and flexible installation. The whole system can easily be moved to different workplaces, where it can then be used as a tool for handling objects. These advantages will be demonstrated in the second half of the ALEXA experiment. Figure 1. shows the robotic arm.

II. Inventions in Design and Construction

After introducing some important constraints, we will show how the link lengths were optimized. Here, the optimization was meant to obtain a maximal workspace while taking the constraints due to the nominal drive torque $\hat{\tau}_i = 5Nm$ and a payload of $m_G = 250\text{g}$ into account. Further, a tensing mechanism and a gravity compensation mechanism will be presented. Finally, the gripper and its novel drive will be described.

A. Joint Constraints

As mentioned, the robotic arm was utilized for common pick-and-place operations. Therefore, it was necessary that it obtains a suitable number of degrees of freedom. Due to constraints from the *robolink* design, the robotic arm was provided with five degrees of freedom, which are embodied by five revolute joints. All joints, with the exception of the the base joint, are *robolink* joints. Two different *robolink* joint types were used – two joints with an asymmetrical single pivot axis ($40° \leq \gamma_{2,3} \leq -130°$), and one type with a synchronous pivot ($|\gamma_4| \leq 90°$) and rotation axis ($|\gamma_5| \leq 270°$). The base joint is a proprietary construction ($|\gamma_1| \leq 180°$).

B. Workspace Optimization

In general, the workspace dimension of the robotic arm depends on the lengths l_i of its joint connecting links $i \in \{1, 2, 3\}$. All link lengths l_i shall be determined for a largest possible workspace given by a specific width D_W along axis $\tilde{x}_0$ of frame $\tilde{\mathbf{K}}_0$ as introduced in Figure 2. Further, the torque of joint 2, that is the highest loaded joint for the most common load cases, should not exceed the nominal torque $\hat{\tau}_2$ of the actuating motor drive. Finally, the optimization must be able to carry the specified payload of 250g and a gripper length of $l_G = 150\text{mm}$.

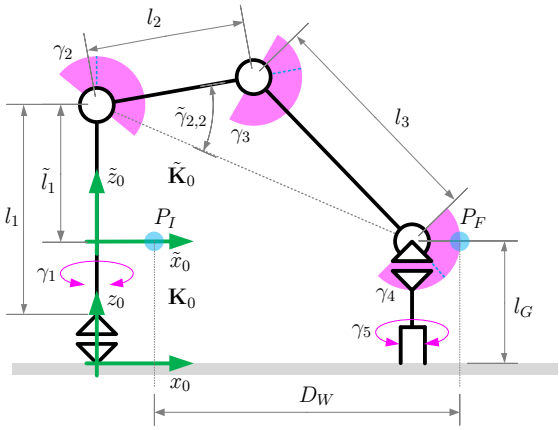


Figure 2. Model of the ALEXA robotic arm for workspace optimization

The determination of the optimal link lengths l_i was achieved by analyzing a simulated pick-and-place operation. In particular, the optimization variables are $\tilde{l}_1$ (length of link

*Research funded and supported by the ECHORD project.

1 minus l_G), l_{23} (total length of link 2 and 3 given by $l_{23} = l_2 + l_3$), and k (ratio of l_2 and l_{23} determined by $k = l_2/l_{23}$). A vector that summarizes the optimization variables is expressed by $\boldsymbol{\xi} = [l_1 \; l_{23} \; k]$. Since the rotary angles of all joints are bounded by $\mathbf{q}^l \leq \mathbf{q} \leq \mathbf{q}^u$ with $\mathbf{q} = q_i = \gamma_i$, the range of $\boldsymbol{\xi}$ must be achieved. First, the *robolink* construction kit requires links with a minimum length of $l_0^l = 200\text{mm}$. Then, the lower boundary for l_{23} is

$$l_{23}^l = 2l_0^l. \tag{1}$$

The upper boundary value l_1^u of link 1 is determined by

$$l_1^l = l_0^l \quad l_1^u = \begin{cases} -R\cos(\gamma_2^l - \tilde{\gamma}_{2,2}) & (\pi - \gamma_2^l) > \tilde{\gamma}_{2,2} \\ R & \text{else,} \end{cases} \tag{2}$$

while R denotes the distance from joint 2 to the wrist joint of the robot given by $R = l_{23}\sqrt{2k(1-k)(\cos\gamma_3^l - 1) + 1}$. Further, the range of k can be also determined by

$$k^l = l_0^l/l_{23} \qquad k^u = 1 - l_0^l/l_{23} \tag{3}$$

In the optimization, the joint torques were computed, which occur during the transportation of the payload from an initial position $P_I(\tilde{x}_I, 0, 0)$ to a final position $P_F(\tilde{x}_F, 0, 0)$ with a constant velocity of $v_E = 0.15\text{m/s}$. By knowing the range of l_1, the x element of P_I and P_F can be computed as follows

$$\tilde{x}_1 = \sqrt{R^2 - l_1^2} \qquad \tilde{x}_2 = \sqrt{l_{23}^2 - l_1^2}.$$

Both coordinates correspond with the inner and outer boundary of the workspace along axis $\tilde{x}_0$, so that the specific width D_W is given by $D_W = \tilde{x}_F - \tilde{x}_I$. Next, a position $\mathbf{p}[t_j] = \mathbf{p}_j$ and velocity $\dot{\mathbf{p}}[t_j] = \dot{\mathbf{p}}_j$ trajectory was computed in the Cartesian frame along $\overline{P_I P_F}$ for the quasi-static case. The time interval is defined as $t_j \in \{t_0, \ldots, t_N\}$ whose edge-times are given by $t_0 = 0$ and $t_N = D_W/v_G$. Next, the workspace trajectories were transformed in the joint space by

$$\mathbf{q}_j = \mathbf{f}(\mathbf{p}_j) \qquad \dot{\mathbf{q}}_j = \mathbf{J}(\mathbf{q}_j)^{-1}\dot{\mathbf{p}}_j \qquad \ddot{\mathbf{q}}_j = \dot{\mathbf{J}}_j(\mathbf{q}_j)^{-1}\dot{\mathbf{p}}_j.$$

Then, the equation of motion gives the joint torques $\boldsymbol{\tau}[t_j]$ of the robotic arm

$$\mathbf{M}(\mathbf{q}_j, \boldsymbol{\xi})\ddot{\mathbf{q}}_j + \mathbf{C}(\mathbf{q}_j, \dot{\mathbf{q}}_j, \boldsymbol{\xi})\dot{\mathbf{q}}_j + \mathbf{g}(\mathbf{q}_j, \boldsymbol{\xi}) = \boldsymbol{\tau}[t_j]. \tag{4}$$

while $\mathbf{g}(\mathbf{q}_j, \boldsymbol{\xi})$ also takes the mentioned payload into account. The torques of joint 2 were separated by

$$\tau_2[t_j] = [0 \; 1 \; 0]\boldsymbol{\tau}[t_j] \qquad \boldsymbol{\tau}_2^T = \begin{bmatrix}\tau_2[t_0] & \ldots & \tau_2[t_N]\end{bmatrix}.$$

that gives the constraint for the actuating motor drive

$$\hat{\tau}_2 \leq u_2\tau_2^{max} \qquad \tau_2^{max} = u_2 \max_{j\in J}\{\boldsymbol{\tau}_2\}, \tag{5}$$

where u_2 denotes the transmission ratio from the joint winch to the motor drive. Next, the cost function, that had to be minimized during optimization, was defined by

$$z(\boldsymbol{\xi}) = D_W^{-1}. \tag{6}$$

while Eq. (1) to (5) yield the related constraints. Finally, Eq. (6) was solved by using the `fmincon` solver of MATLAB. The result are $\tilde{l}_1 \approx 0.35\text{m}$, $l_{23} \approx 0.8\text{m}$, and $k = 0.44$. This gives a maximal torque of $4.94\,\text{Nm}$ on the motor side with a safety coefficient of ≈ 2. Further, a specific workspace width of $D_W = 0.7\text{m}$ is given.

C. Tensing Mechanism and Gravity Compensation

A single robolink joint is actuated by two ropes in an antagonistic manner. Hence, there is always a tight side of the rope and a slack side. In general, the slack side of the rope also causes relaxation of the related Bowden cable, which in turn can result in the rope slipping off the winch. To avoid this situation, a tensioning mechanism was investigated, as shown in Figure 3. {A}. It consists of a sleeve {1}, a screw {2} with a through-hole, and a spring {3}. The end of one Bowden cable {4} is plugged into the upper side of the sleeve. The lower side of the sleeve is plugged into the screw against the spring, which is mounted inside. Finally, the assembly is mounted in a threaded hole of a sheet, with the spring prestressed as the rope, which leads from the Bowden cable through all parts to the motor driven winch, is tightened. In this arrangement, the rope {5} is tightened in any case, given the range of the spring is sufficiently dimensioned according to the amount of slack to be expected.

As written in the last preceding section, joint 2 has the highest load. In spite of the mechanical dimensioning of the robot structure, the required motor torque exceeded the nominal value due to a higher influence of sliding friction in the Bowden cables as it was assumed. For that reason, a compression spring {6} with high stiffness was used to compensate the weight force, which caused the major load of joint 2. Here, the spring gets tensed with joint 2 moving in a negative rotational direction, when the robotic arm moves downwards. Alternatively, the tensed spring works with the motor of joint 2 when the robotic arm is lifted up again.

D. Gripper

In the experiment, we designed a new radial gripper that is actuated by Bowden cables. As Figure 3. {B} shows, the gripper was equipped with two compliant FinRay® fingers to provide compliant gripping capabilities.

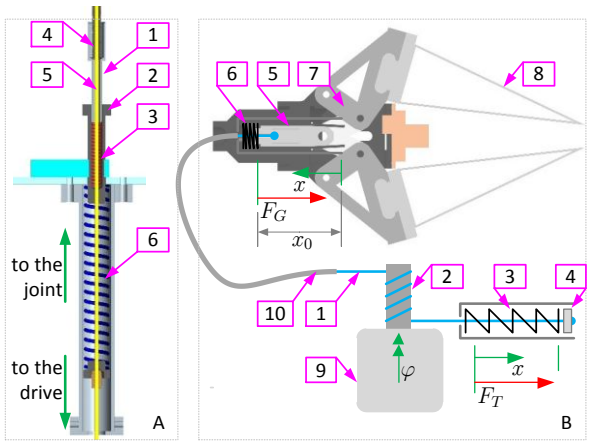


Figure 3. Rope tensioner and gravity compensator {A}, *d*rive system of the gripper {B}

The gripper has a novel drive that limits the gripping force in a unique and simple way. The actuating rope {1}

leads from a tension mechanism through a motor driven endless winch {2} to the gripper. The tensioner consists of a compression spring {3} and a piston {4}, while the rope is attached to the said piston. Further, the piston is also mounted on the spring, whose stiffness is denoted as c_T. By pulling the rope, the spring gets tensed and caused a force F_T that acts against the piston. According to the introduced direction x, the force is $F_T = c_T x$. Within the gripper, the other end of the rope is connected to a stamp {5}, which is mounted in a shaft on a second spring {6}. Further, the stamp actuated the mechanical linkage {7} that rotates the fingers {8}. Here, the stiffness of the spring inside the gripper is denoted as c_G. The resulting force that acts on the stamp is F_G. In particular, it consists of the spring force as well as the force that is caused by the gripped load $F_G = c_G(x_0 - x) + F_L$.

For closing the gripper, the motor drive {9} must rotate in negative direction $\dot{\varphi} < 0$, whereby the rope is pulled in positive direction $\dot{x} > 0$. Due to the friction on the endless winch, the rope is pulled as long as the following inequality is satisfied

$$k_B F_G < k_T F_T \quad \rightarrow \quad \dot{x} > 0 \tag{7}$$

where k_B, k_W denote the force gain due to friction in the Bowden cable {10} and on the winch. When force equilibrium is reached, the gripper is blocked by its mechanical stops or an object that was gripped successfully. Both cases leads to $\dot{x} = 0$. If the winch rotates in positive direction $\dot{\varphi} > 0$, the gripper opens and the tensioner gets tensed again as long as the inequality

$$F_G/k_B > F_T/k_T \quad \rightarrow \quad \dot{x} < 0 \tag{8}$$

is satisfied.

III. Control of the Robot System

The control topology of the ALEXA robot consists of a path planner, a trajectory generator and five position controllers. All control components were optimized for a cable-driven robot. In this chapter, an overview of the main achievements is given.

A. Computation of Torque Minimized Trajectories

The main control of the robot has different capabilities to generate paths between an initial position $\mathbf{q}_I$ and final position $\mathbf{q}_F$. Depending from the specific task, the path can be either calculated as a minimized joint load path or as a simple LIN path. However, every trajectory is constrained in case of end effector velocity $\mathbf{v}_E$, angular velocity $\dot{\mathbf{q}}$, acceleration $\ddot{\mathbf{q}}$, and jerk limitation $\dddot{\mathbf{q}}$

$$\|\mathbf{v}_E\| \leq v_E^{max} \quad |\dot{\mathbf{q}}| \leq \dot{\mathbf{q}}^{max} \quad |\ddot{\mathbf{q}}| \leq \ddot{\mathbf{q}}^{max} \quad |\dddot{\mathbf{q}}| \leq \dddot{\mathbf{q}}^{max} \tag{9}$$

By satisfying these constraints, smooth motions can be ensured while reducing the vibration stimuli of the elastic joint-rope-motor system. Furthermore, the end effector is adjustable to a harmless value in case of human robot safety.

The entire path planning process is separated in three steps. First, a path is computed that avoids all obstacles and yields minimized joint torques for the static case ($\dot{\mathbf{q}}, \ddot{\mathbf{q}} = 0$). This will be achieved by using discretized torque maps as shown in Figure 4. In particular, these maps present the allocation of the joint loads (depending on the joint configuration).

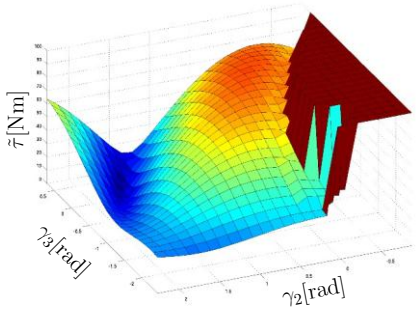


Figure 4. Discretized torque map for path optimization

The torque maps were determined by placing discretized joint positions $\Delta\gamma_i$ into the equation of motion Eq. (4) of the loaded robot

$$\mathbf{q}(\boldsymbol{\kappa}) = \Delta\gamma\boldsymbol{\kappa}^T \qquad \boldsymbol{\kappa} \in \mathbb{N}^{1\times 4} \tag{10}$$

where $\kappa_i \Delta\gamma$ is within the range of the related joint i. Next, the vector norm of $\boldsymbol{\tau}$ is allocated to a unique joint configuration. Since the position of joint 5 has no effect on the torque norm, it is ignored during the map determination. Afterwards, the map is updated to take collisions into account. By analyzing the DH matrices to the joint frames, ground and self-collisions are detected as well as collisions with obstacles in the workspace. Every collision is marked with an extremely high torque denoted by τ_∞.

In the next step, based on the map, a path with a number of N_C joint configurations, which are torque minimal, is obtained by solving the following optimization problem

$$\min_{\boldsymbol{\kappa}_j, N_C} \sum_{j=1}^{N_C} \tilde{\tau}(\boldsymbol{\kappa}_j) \rightarrow \mathbf{Z} \qquad \mathbf{Z} = [\boldsymbol{\kappa}_1 \ldots \boldsymbol{\kappa}_{N_C}]$$

where the constraints are given by

$$\mathbf{q}^l \leq \mathbf{q}(\boldsymbol{\kappa}_j) \leq \mathbf{q}^u \qquad \tilde{\tau} < \tau_\infty$$
$$\mathbf{q}(\boldsymbol{\kappa}_1) = \mathbf{q}_I \qquad \mathbf{q}(\boldsymbol{\kappa}_{N_C}) = \mathbf{q}_F$$

Due to the discretization of the torque map, the path optimization yields a set with a number N_S of single path segments.

In the last step, constrained motions are computed for these segments that are achieved by using an algorithm for PTP motion generation as presented next. First, the joint $\hat{\imath}$ with the highest angle difference between its initial and final position is selected. For synchronous motion, the velocity of all other joints are made dependent on $\dot{q}_i$

$$\mathbf{q}_F - \mathbf{q}(t) = \{q_{F,\hat{\imath}} - q_i(t)\}\mathbf{v} \qquad \dot{\mathbf{q}}(t) = \dot{q}_i\mathbf{v} \tag{11}$$

For smooth acceleration and deceleration motions, the joint position trajectories were determined by a 5$^{\text{th}}$ order polynomial approach. Here, all boundary values of the polynomial are given by Eq. (9) and (11). Satisfying the boundary values ensure that the end effector reached its maximal velocity or at least one joint reached its maximal angular velocity

when the acceleration phase, which is the first of three phases, is finished.

In the second motion phase, the velocity is kept constant. Here, the time is discretized by $t_j = \{t_0, \ldots, t_N\}$ where t_0 denotes the moment at which the acceleration phase ends and t_N the moment at which the phase of constant velocity ends. According to the recent active constraint, it is

$$\dot{q}_i[t_j] = \begin{cases} \dfrac{v_E^{max}}{(t_j - t_{j-1})\|\mathbf{J}(\mathbf{q}_{(j-1)})\mathbf{v}\|} & v_E \geq v_E^{max} \\ \dot{q}_i[t_{j-1}] & \text{else} \end{cases}$$

This process is repeated until a position is reached at which the highest deceleration $\ddot{q}_i[t_j] = \ddot{q}_i^{max}$ is needed to stop joint $\hat{\imath}$ in its temporary final position $q_{\hat{\imath},\hat{\jmath}}$ of the current path segment $\hat{\jmath} \in \{1, \ldots, N_S\}$

$$\ddot{q}_i[t_j] = \frac{\dot{q}_i^2[t_j]}{2q_i[t_j] - 2q_{\hat{\imath},\hat{\jmath}}}$$

Next, the acceleration or deceleration phase of the following path segment is determined.

B. Joint Position Control

An independent joint control approach was used to control the robotic arm. The control input of the controller is the motor velocity $\dot{\varphi}_i$ with the motor angle φ_i and the joint position q_i as feedback variables. A simulation of the elastic motor-rope-joint system, which also includes nonlinear friction model, shows, that the friction force $\mathbf{f}_R \in \mathbb{R}^{1\times 5}$ and the elasticity $\mathbf{E} = \mathrm{diag}(\epsilon_i)$, of the system generates a large backlash $\boldsymbol{\varphi}_B$ when the motors changes its rotation direction. The backlash for all joints is determined by $r\boldsymbol{\varphi}_B = \mathbf{E} \cdot \mathbf{f}_R$, where r denotes the radius of the rope winch.

For example, joint 3 has a backlash of $\varphi_{B,3} = 120°$ that needs about 2s to compensate by rotating motor 3 with maximal speed. Agrawal et al. [3] presented different approaches for backlash compensation. Further, Jacobson et al. [4] defined control laws for tendon driven robots. In particular, a cascade controller with compensators is discussed. However, the system presented here is stronger influenced by friction and elasticity. Hence, the use of a common P- or PI-controller will obtain slow dynamic behavior and heavy overshooting effects, as the simulation shows. In particular, the controller must be heavy "detuned" to stabilize the system.

A combination of a P-controller with motor position feedback φ_i and an I-controller with joint position feedback was investigated. The controller output is given by

$$\dot{\boldsymbol{\varphi}} = \mathbf{K}_P(\boldsymbol{\varphi} - \boldsymbol{\varphi}^*) + \mathbf{K}_I \int \mathbf{q} - \mathbf{q}^* \, dt$$

As shown by Figure 5. , this approach provides a much better control behavior than a common one. Further, the set-point trajectories of the motor are determined by

$$\boldsymbol{\varphi} = \mathbf{E}\boldsymbol{\tau} + \mathbf{U}\mathbf{q},$$

where $\boldsymbol{\tau}$ is given by the EOM of the robot. Further, $\mathbf{U} = \mathrm{diag}(u_i)$ denotes the ratio matrix, which is given by the ratio of the joint and motor winch. To avoid wind-up effects, the integrator output is bounded, while these boundaries correspond with the maximal backlash caused by friction. Here, the superior control set-point is the motor trajectory that yields stability for all $\mathbf{K}_P$ in the closed-loop. The optimal controller parameters where found in an iterative process.

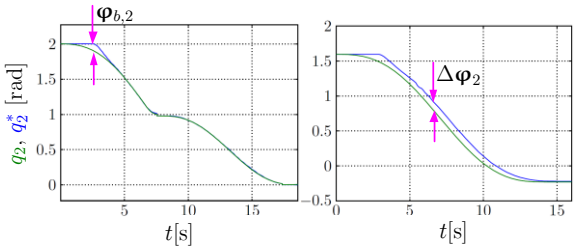


Figure 5. Set-point tracking of the combination controller (left) and the P-controller (right)

The combination controller has good set-point tracking capabilities and less overshooting effects, as Figure 5. shows. Unfortunately, the big backlash, especially at the beginning of the motion, causes asynchronous joint movements and a big tracking error as well, which could result in collision. Hence, a prestressing control was investigated as presented in the next section.

C. Prestressing Control

The prestressing controller assumes that the backlash is related linearly to the joint torques, the direction of the last joint motions $\mathrm{sgn}\{\dot{\mathbf{q}}(t_{-1})\}$, and the planned joint motions $\mathrm{sgn}\{\dot{\mathbf{q}}(t)\}$. The control law is defined by

$$\boldsymbol{\varphi}_B = \tfrac{1}{2}(\mathrm{sgn}\{\dot{\mathbf{q}}(t_{-1})\} - \mathrm{sgn}\{\dot{\mathbf{q}}(t_{-1})\})(\mathbf{K}_\tau\boldsymbol{\tau} + \boldsymbol{\varphi}_{B0})$$

Before a motion starts, every motor rotates until the predicted backlash value is compensated. An upcoming change of the friction and the backlash offset is reduced by an adaptive adjusting of the friction matrix $\mathbf{K}_\tau$ and the backlash offset vector $\boldsymbol{\varphi}_{B0}$ constantly. Here, the joint behavior is monitored and analyzed during the prestressing process. A adaptive adjustment law yields an update for $\mathbf{K}_\tau$ and $\boldsymbol{\varphi}_{B0}$.

REFERENCES

[1] G. O. Young, "Synthetic structure of industrial plastics (Book style with paper title and editor)," in *Plastics*, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.

[2] S. Klung et al. "Design and Control Mechanisms for a 3 DOF Bionic Manipulator" in: Proc. 1st IEEE / RAS-EMBS Intl. Conf. on Biomedical Robotics and Biomechatronics (BioRob). Pisa, Italy, 2006

[3] V. Agrawal et al., "Control of Cable Actuated Devices using Smooth Backlash Inverse" in 2010 IEEE International Conference on Robotics and Automation pp. 1074–1079

[4] S. Jacobson et al. "Control strategies for tendon-driven manipulators." in Control Systems Magazine, IEEE , Vol.10, No.2, pp.23-28, Feb 1990

Mobile Robotics on Construction Sites: Dimensional Tolerance Handling, dimRob*

Volker Helm, Selen Ercan, Fabio Gramazio, and Matthias Kohler

***Abstract*— In this paper, viable applications of mobile robotic units on construction sites are explored and accordingly, potential areas in the building sector are identified, with the intention being to build upon innovative man-machine interaction paradigms to deal with the imprecision and tolerances often faced on construction sites. By combining the precision of the machine with the innate cognitive human skills, a simple but effective mobile fabrication system is experimented for the building of algorithmically designed structures that would not be possible if left to conventional manual methods. It is assumed that this new approach to in-situ construction aimed at a deeper integration of human ability with the strengths of digitally controlled machines, will result in advances in the construction sector, thus opening up new design and application fields for architects and planners.**

I. Introduction

The technical requirements of building processes are growing increasingly complex, and as such necessitate custom solutions. Computer-controlled pre-fabrication methods offer one such solution for the fast and cost-effective manufacture of customized physical components. So far, construction processes have been mainly handled statically. In a new approach to tomorrow's building requirements, it is proposed that a generic mobile fabrication unit (Fig. 1) can be employed directly on a building site to carry out tasks that have previously been the realm of off-site specialized stationary machines in pre-fabrication. A combination of the benefits of digital manufacturing and in-situ construction offers an interesting alternative to the conventional approach of pre-fabrication of building elements. Those benefits include: (1) when programmed to handle local specificities and tolerances, a robotic unit will be able to respond to the uniqueness of each construction site, bringing flexibility to the process, (2) while at the same time removing the need for the costly and unsustainable transportation of large building elements.

It was back in 1990s that the research on automation and robotics in construction could be listed as; views for automation and robotics in construction; design, construction planning and management; elemental technologies for automation and robotics; and application [1]. Some construction tasks that come forward regarding the application of the research are: Masonry construction, assembly and finishing operations, surveying and positioning,

*Research supported by the ECHORD (European Clearing House for
V. Helm, S. Ercan, F. Gramazio, M. Kohler are with Architecture and Digital Fabrication, DFAB, at Swiss Federal Institute of Technology (ETHZ), Zurich, Switzerland (+41 44 633 4906; helm@arch.ethz.ch, ercan@arch.ethz.ch, gramazio@arch.ethz.ch, kohler@arch.ethz.ch).

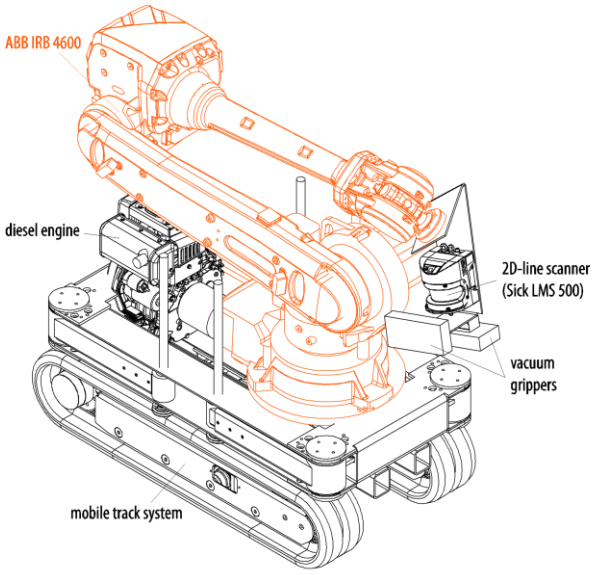


Figure 1. The fabrication unit: An already exsiting industrial robot, ABB IRB 4600, mounted on an engineered compact mobile track system that is sized to fit through a standard door frame on a construction site. Engineered and integrated parts are shown in black.

inspection, repair and maintenance, material handling, and concrete placing and finishing [1]. Attempts in the early 1990s to replace manual technologies with robotic building processes in the construction sector resulted in such automated brick-laying research projects as ROCCO in 1994 [2], ESPRIT in 1995 [3] and BRONCO in 1996 [4]. While automated bricklaying - masonry construction - projects constituted 6% [5] to 7% [1] of the applied research work into robotics on construction sites, they have yet to be put to practical use. The main reasons for this were, firstly, the machines could not compete with manual labor as they were not economically feasible, which constitutes a unique research area on its own. And secondly, they did not fully address the uniqueness of the building site and the dimensional tolerances of the task in hand. For justification of automation in robotics on construction sites, it is clear that, the process of construction has to be financially proved even though it is not the concern of the current state of this research since the aim is rather than developing an automated robotic device, it has been to design a robust system that takes full advantage of the specific complementary strengths of both man and machine.

Robotic fabrication has traditionally been embedded within the high-tech industrial environment, where fixed positioning and constant conditions determine the role that the robot may undertake in the fabrication process. Unlike in pre-fabrication facilities, a construction site is a spatially

Figure 2. The compact mobile track system and the fabrication unit.

complex and heterogeneous environment, where a robotic system may be exposed to continuous change and unpredictable events. For this reason, a fabrication unit needs to be made aware of its surroundings, its components and its own position, as well as the tolerances generated by inaccuracies in the materials with which it is working. Therefore, the methodology of this research involves the use of scanning systems and object recognition technologies that can be simplified through human instruction.

Rather than the development of a new, advanced piece of machinery, the research expands on different options of digital manufacturing in the construction sector, defining future application areas for mobile architectural robotics and the paper is organized as follows: After the setup is introduced, the 3 specific demonstrations are explained, holding the experiment objectives: (1) 2D line scanning different-sized building elements for handling of tolerances; (2) 3D scanning hand movements for implementing man-machine interaction; (3) and 3D scanning of local reference markers in an unknown workspace for the mobility of the fabrication unit.

II. PROOF OF CONCEPT

In this research, the responsibility of decision-making is shared between the human and the machine, whereby the machine autonomously solves problems that cannot be efficiently addressed by its human counterpart. In this way, the mobile unit, comprising an existing industrial robot mounted on a mobile base, is able to fabricate algorithmically generated structures in-situ, employed in real-world contexts such as the construction site.

For the application of this experimental process, firstly, the ABB IRB 4600 was selected for the experiment and it was mounted on a compact mobile track system that was sized to fit through a standard door frame on a construction site in its folded position. Each individual component listed in the experiment setup was integrated cohesively into the system enabling a high degree of flexibility.

A. *Experiment Setup*

Initial infrastructural phase of the experimentation process consists of assembling a mobile fabrication unit for employing it in real-world contexts in addition to stationary conditions. This phase includes;

- A market survey on selecting a suitable robot through which; the ABB IRB 4600, providing a wide operational range, was found best fitting compact robot to the requirements of the project. Holding the properties; a low weight and a high loading capacity; it was suitable for constructing elongated high structures, without being adjusted, using a variety of building elements.
- Design and engineering of a mobile track system, supported by side-hinged telescoping outriggers with integrated raising jacks to ensure stability together with mobility. The unit is driven by a diesel engine attached to the base, which within the scope of this project only utilizes/provides the mobility of the system. The engineering of the base is driven by the ideal workspace of the fabrication unit, that is the construction site, in addition to the calculations regarding the moment and the center of gravity.
- Integration of different scanning systems to the fabrication unit, serving different objectives of the experiment being: (1) A 2D-line scanner (Sick LMS 500) for detection of dimensional tolerances (2) and a 3D scanner enabling the detection of objects or obstacles, etc. on the workspace. The 3D scanner also facilitated the communication of the unit with the innate cognitive skills of the human counterpart for the supervision in the self-orienting system developed for mobility.
- Integration of two vacuum grippers to the fabrication unit in different positions on a multifunctional tool enabling the unit to grip building elements either from the top or from the side to apply various additive fabrication strategies. This facilitated particular additive construction tasks involving gripping individual building elements at high speed and placing them with an effective precision. The vacuum pressure is created by a vacuum pump that is installed at the back of the robot in order to have a compact mobile unit.

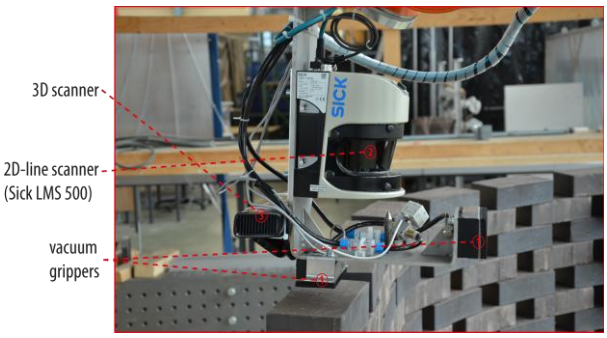


Figure 3. The work tool consists of two vacuum grippers, a 2D-line scanner and a 3D scanner.

B. Application

This part constitutes the main body of the methodology, where the concept is proven by demonstrations in 1:1 scale (including the progress also on the software side to facilitate fabrication).

Handling of Tolerances: Throughout the research process, various experimentations took place regarding the needs of the construction tasks. These include a reactive fabrication system requiring the scanning of different-sized wooden building elements. With this system, the indeterminacy of materials used in the fabrication process can be mapped as every operation on a building site is unique in terms of the dimensions of the materials used and the range of tolerances. A 1:1 scale building process was demonstrated at 2011 Fabricate Conference in London as part of the experimental application process of the mobile fabrication unit prior to its deployment on an actual construction site. In this process, the reactive fabrication system is tested based on a feedback strategy. For this 1:1 scale test, an articulated timber formation was assembled from 1,330 wooden building blocks of three different thicknesses simulating the range of dimensional tolerances that would be faced on the construction site. During fabrication, after laying the elements, the fabrication unit maps the indeterminacy accumulated by various dimensions of the bricks placed side by side, by scanning each layer (Fig. 4). With fast and accurate measurement, it is handled autonomously by the robot, which is a requirement that cannot be addressed efficiently by the human counterpart. The mapped measurement is then sent back to the design/control software, and the robot arm re-orientates itself according to the new set of height and angle data.

Man-Machine Interaction: As the fabrication unit is

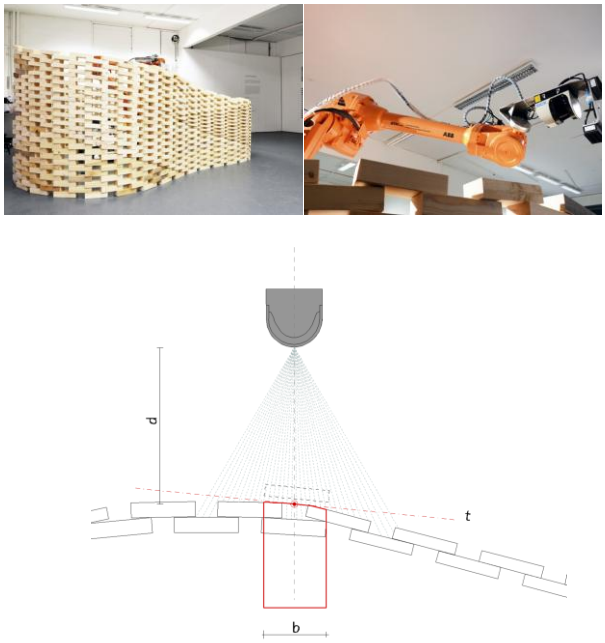


Figure 4. After laying individual wooden building blocks with different thicknesses, the fabrication unit can map the indeterminacy by scanning each layer. A 1:1 scale building process was demonstrated at 2011 Fabricate Conference in London.

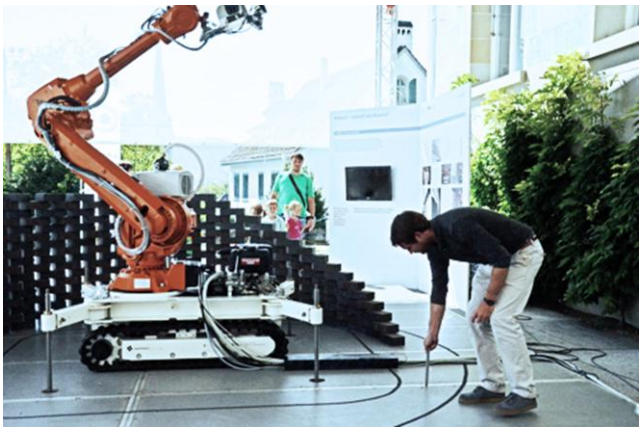


Figure 5. A 1:1 scale fabrication process facilitated by man-machine interaction was demonstrated at the 2011 Scientifica Exhibition in Zurich.

intended to be employed directly on a construction site, it is required to operate within unknown contexts and workspaces, and must orient itself while recognizing objects, obstacles, workers, etc. in its surroundings. Similar to the experimentation on handling of tolerances, a 1:1 scale fabrication cycle applying an interactive process was demonstrated at the 2011 Scientifica Exhibition in Zurich (Fig. 5). In this demonstration, the audience participation was employed efficiently as the human supervision for which the 3D scanning technology implemented detects and processes the movements of the hand, and in this way the human collaborator was able to "show" the robot its working area and building zone in reference to its relative position. For this, a feedback loop was programmed that provides a continuous communication between the robot controller and the scanning system. This continuous exchange of information feeds the fabrication process with hand movements that are scanned and imported as CAD data to the design software. This data is then converted into robot code, by which the fabrication unit is informed in real time and builds accordingly.

Mobility: For the proof of concept regarding the direct use of robotic units on construction sites, this phase of the methodological process, being the mobility, is especially significant in terms of bringing the robotic operational capabilities from stationary conditions to the production of a complex architectural component in-situ. For this, a system is developed for self-positioning of the mobile unit through the 3D scanning of local points (such as the center point of a metal disk – the satellite) set by the human collaborator in the workspace (Fig. 6). By scanning these points, the fabrication unit is able to move several times, relocating itself according to the local reference system to facilitate the fabrication process of large building components in separate sections. The modular 8-meter wall in Fig. 7, is assembled by several consecutive self-positioning of the mobile fabrication unit in the parking garage of the Department of Architecture at ETH Zurich whose conditions are experimented as that of a real construction site. The scale and modularity of the wall structure were directly derived from these surrounding spatial conditions for instance one being the ceiling height while several aspects of the design were driven by the selected robotic fabrication method. Those include the operational

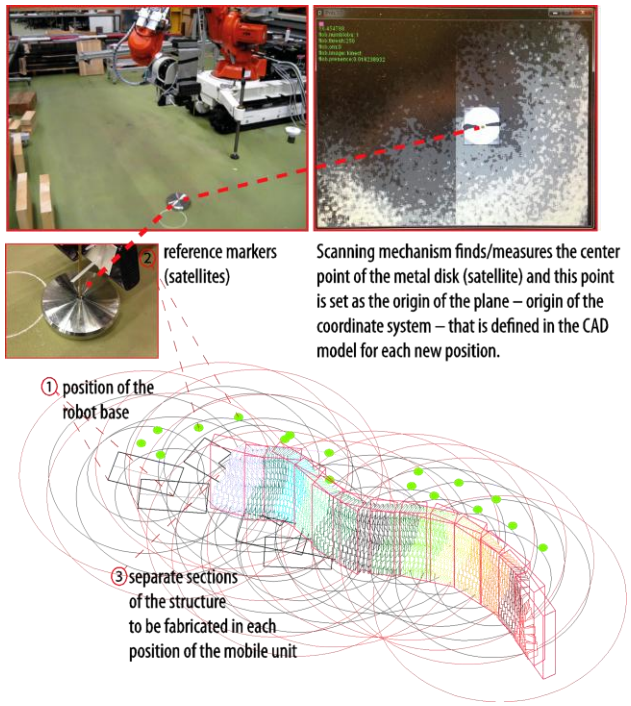


Figure 6. Diagram showing the partitioning in the overall fabrication strategy and the scanning system and the each new position of the fabrication unit with circular lines indicating the operation range of the robot arm.

range of the robot arm which have defined the positioning of the fabrication unit for ascertaining an optimum partitioning in the overall fabrication strategy (Fig. 6). These are all tested and simulated on the software side via offline programming prior to a 1:1 scale experimental fabrication, including the testing of a system which eases the export from the CAD model to the robot controller.

III. Conclusion

The work described in this paper is intended as a first step in the evolution of mobile robotics on construction sites. In contrast to previous researches, the fabrication unit described in this experiment is intended as an open system that is adaptable to different applications and situations, and accordingly, the main objective of this research has been to identify different application scenarios and to illustrate various communication and data acquisition systems. With this paper, a fabrication system is proposed that can respond flexibly to the rapidly changing requirements of the construction sector and can close the gap between the planning and in-situ production of a large, non-standard and complex architectural component.

Acknowledgment

The authors would like to thank the industrial partner Bachmann Engineering AG for its commitment in engineering the experimental setup. We are also grateful to ABB for its generous support to the project. Most of the research application would have been impossible without the valuable support of all the members of DFAB, Architecture and Digital Fabrication Chair, ETHZ, of Prof. Fabio Gramazio and Prof. Matthias Kohler. Part of the research presented in this paper is a collaboration between dimRob and an elective course ("The Fragile Structure") led by Luka Piskorec at the chair of Architecture and Digital Fabrication. The students were: Petrus Aejmelaeus Lindstrom, Leyla Ilman, David Jenny, Michi Keller, Beat Ludi.

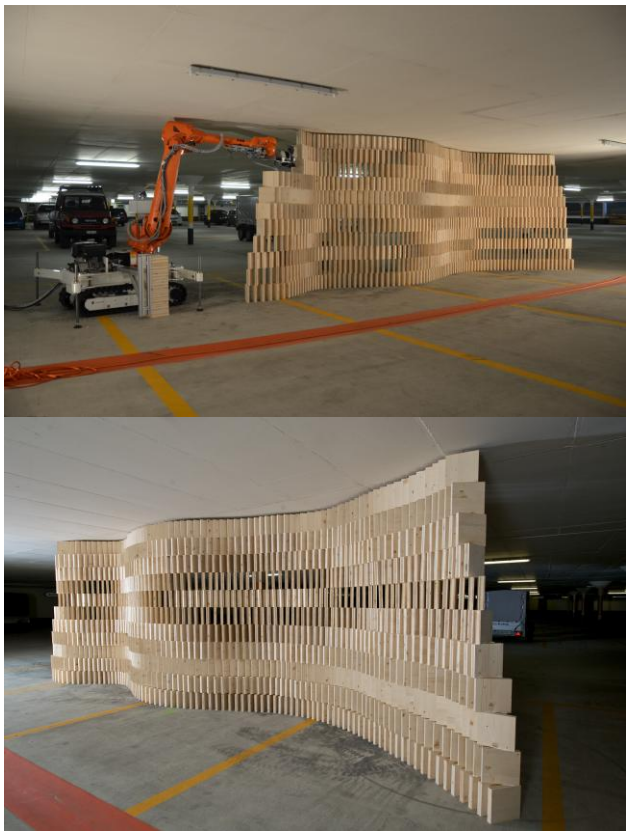


Figure 7. The mobile fabrication unit re-positioned itself several times during the fabrication of this 8-meter modular wall in the parking garage of the Department of Architecture at ETH Zurich. The unit was designed by the students of an elective course ("The Fragile Structure") under the professorship of Gramazio & Kohler.

References

[1] T. Ueno, "Trend Analysis of ISARC," in *Proc. 15th ISARC International Symposium on Automation and Robotics in Construction*, Munich, 1998, pp. 5–12.

[2] J. Andres, T. Bock, and F. Gebhart, "First results of the development of the masonry robot system ROCCO," in *Proc. 11th ISARC International Symposium on Automation and Robotics in Construction*, Brighton, 1994, pp. 87–93.

[3] T. Bock, T. Linner, W. Lauer, N. Eibisch, "Automatisierung und Robotik im Bauen," in *Arch+ Zeitschrift für Architektur und Städtebau*, Mai 2010, pp. 34–39.

[4] G. Pritschow, M. Dalacker, J. Kurz, and M. Gaenssle, "Technological aspects in the development of a mobile bricklaying robot," *Automation in Construction*, vol.5, no.1, pp. 3–14, 1996

[5] A. Warszawsk, R. Navon, "Survey of Building Robots," in *Proc. 13th ISARC International Symposium on Automation and Robotics in Construction*, Tokyo, 1996, pp. 205–211.

ROS-Industrial – Applying the Robot Operating System (ROS) to Industrial Applications*

Shaun Edwards, Chris Lewis, *Member, IEEE*

***Abstract*—This paper describes an ongoing effort to by Southwest Research Institute, Yaskawa Motoman, and Willow Garage to leverage the explosion in machine perception applications developed under ROS toward industrial automation. The effort has developed a number of wrappers enabling ROS to interface directly to industrial robot controllers and other sensors. A "pick and place" application initially developed for the PR2 was ported to and demonstrated on a Yaskawa Motoman SIA10D industrial robot. The ease in which the industrial robot is programmed using ROS to operate in an unstructured and dynamic environment demonstrates that ROS may represent a disruptive technology for industrial automation. A consortium is being formed to fund and continue the development of the ROS-Industrial initiative.**

I. Introduction

Robotic manipulators and robotic vehicles have been deployed in a variety of industrial automation applications for many decades. Because these systems are flexible and reconfigurable they have a cost advantage over custom automation solutions, especially when a limited number of operational units are required. Although there are exceptions, most successful robot applications require the robot to do little more than traverse pre-defined paths while performing repetitive operations. These robots do not adapt to dynamic environments. For example, a manipulator programmed to palletize boxes only process boxes within certain size and shape constraints. Similarly, a robot programmed to paint parts has to be re-programmed for each type of part.

Advances in sensor systems including laser scanners and RGB-D cameras along with associated advances in algorithms provide the opportunity to combine machine perception with adaptive automation solutions to make truly flexible systems capable of processing a wide variety of parts and products flowing simultaneously through a production line. Unfortunately, because of limited resources, concerns about intellectual property, safety and reliability, individual robot manufactures do not include the necessary features in their controllers to easily integrate new perception technologies. For example, a researcher might develop a machine vision algorithm which observes random sized boxes arriving on a conveyer and automatically decides how best to palletize them. Another researcher might develop algorithms to automatically plan paths to paint random and odd shaped parts passing through a paint booth. However, the implementation of these two automation solutions on commercial robot controllers would be difficult due to limited software development tools and missing hardware interfaces for new sensors. Common capabilities for list and numerical processing are simply not available within many industrial robot programming environments [1]. It is not cost effective to implement complex algorithms on these devices. Historically, advanced algorithms developed by academic and other research organizations do not make it into commercial practice due to the intricacies necessary to integrate their capabilities on commercial robot controllers.

By sponsoring the ROS-Industrial initiative, Southwest Research Institute, Yaskawa Motoman and Willow Garage are attempting to bridge the gap between commercial industrial automation and the state of the art research performed at academic and research labs.

II. Background

A. Robot Operating System (ROS)

ROS is a set of open-source software libraries and tools developed by Willow Garage designed to help developers create robot applications [2]. Its ease of use and impressive array of both hardware drivers, and software capabilities has spawned a rapidly growing community of software developers. ROS uses a publish-subscribe architecture to orchestrate inter-process communication. It has revitalized openCV for image processing. It has created a state-of-the art library for processing point clouds (PCL) so that developers may make use of recent very inexpensive 3D sensing technologies [3]. One of the keys to its success is the uniform way in which packages are built, documented, and in which interfaces are defined. Developers all over the world are publishing packages which can be downloaded, quickly compiled and tested by the community using the exact same code and sample sets. This allows side-by-side comparison of competing algorithms, as well as crowd-sourced testing of complex algorithms. Such capability can be described as disruptive within the robotics research communities.

Willow Garage introduced ROS to a critical mass of developers by distributing a custom service robot, the PR2 [4], to many of the top research organizations throughout the world. Together with Willow Garage, these developers demonstrated the utility of the ROS framework and spawned an explosion of cooperative development. For example, ROS has been used to fly unmanned airplanes, drive unmanned vehicles, and automate humanoid robots [5]. Unfortunately, most of the work done was academic in nature and is not

*Research supported by Southwest Research Institute.

S. Edwards is with the Southwest Research Institute, San Antonio, TX 78238 USA (e-mail: sedwards@ swri.org).

C. Lewis is with the Southwest Research Institute, San Antonio, TX 78238 USA (e-mail: clewis@ swri.org).

easily applied to industrial robots due to the reasons we discussed in the introduction.

Southwest Research Institute develops robotic technologies for a number of commercial and government agencies. We began using ROS in Industrial settings about 2 years ago. Due to its many benefits, we will describe the integration efforts underway to help ROS become as ubiquitous with the industrial automation community as it is becoming in the academic research community.

III. Industrial Integration of ROS

A. Standardizing Industrial Hardware Interfaces

Flexible automation systems often include independent motion control devices, programmable logic controllers (PLCs), and a variety of sensors and actuators. Most of these devices, including most industrial robots, adhere to one of many standard communication interfaces with an exposed application programming interface (API). Because of this, most industrial devices have the potential to become a ROS capable device, to participate in a ROS network and to take advantage of the growing body of technologies for perceptive reasoning available in the ROS community. Initially, the ROS-Industrial program is defining a uniform set of messages so that software wrappers may be developed which transform common industrial devices into ROS capable nodes. Ultimately, we hope that future interfaces to these devices will be designed to be ROS compatible rather than proprietary so that these wrappers are no longer necessary. For example, many servo controllers have an Ethernet communications interface that accepts a set of proprietary commands. For now, our software wrappers consists of a ROS node its associated ROS messages and services which duplicate the proprietary interface. In this case, the ROS node acts as in intermediary that translates messages between ROS and the proprietary messaging protocol allowing other ROS nodes on the network to subscribe to any data coming from the device or to command the device. Visual servoing is an example application where a ROS wrapped servo control mechanism would be combined with a node for a machine vision camera.

B. Industrial Robot Controller Interfaces

The initial hardware focus of ROS-Industrial is to develop the previously mentioned wrapper nodes for industrial robot controllers. While each robot system has a proprietary OS and programming language many commonalities exist.

All robot controllers to varying degrees support socket communications over Ethernet. Libraries for socket communications were developed which transfer data between the ROS wrapper node and the robot controller. While ROS messages may be variable in length, and have complex structure, robot controllers typically have very limited capabilities in dynamic memory allocation, string parsing and memory storage. Therefore, a byte oriented messaging protocol with fixed size message types was implemented between the wrapper node and the robot. Standard messages, expected to be common to most robots, should be easily parsed by different vendor controllers with standard software options. Custom software on the each robot's controller is written to monitor the communications link to the ROS node and to respond to commands and queries. By focusing on standard robot controller software options, interface development need not be performed by robot vendors, allowing the ROS-Industrial community to develop interfaces on their own.

While communications is easily achieved across the spectrum of controller platforms, vendors limit the type of motion allowed through remote interfaces in different ways. Vendors commonly support two modes of operation: trajectory downloading, where a set of waypoints is downloaded to the controller and then executed, and trajectory streaming where waypoints are streamed to the controller as they are executed. Trajectory streaming is not as broadly supported as downloading. As a result, the performance of this interface is limited for those systems. Some controllers cannot execute streamed points at full speed and still guarantee smooth motion. The primary reason for this is not due to latency, rather it is due to the internals of how a controller smoothes a path from point to point. To date, ROS-Industrial interfaces and standard messaging have been developed for the Motoman DX100 and Adept SmartController CX robot controllers.

IV. Benefits

The immediate benefit of the ROS-Industrial program is enabling advanced capability demonstrated by ROS and the PR2 robot on industrial platforms. These capabilities include 3D/2D perception and collision aware path/grasp planning (See Figure 1.)[6]. A "pick and place" application, developed for the PR2 [7] using a Microsoft Kinect 3D sensor [8], was ported to and demonstrated on a Yaskawa Motoman SIA10D industrial robot. The application autonomously identified objects on a tabletop then planned and executed grasping and manipulation operations. Despite the obvious differences between a PR2 and an industrial robot, the majority of the application level software was identical for these two systems. The ability to perform "pick and place" tasks in dynamic and unstructured environments is not a standard capability on robot controllers. The ease in which this capability is achieved within ROS represents a potential for a disruptive change to the industrial robot market (Figure 2.

Providing ROS compatibility between industrial automation devices removes key hurdle in transitioning research to applications. ROS is being widely adopted by researchers, and is becoming the standard platform on which academic robotics research is performed. Through ROS-Industrial, cutting edge research can be immediately deployed to manufacturing applications.

By leveraging an open source solution and providing a consistent build environment with consistent communication standards the ROS-Industrial program has the potential to increase the marked application space and significantly decrease the cost of integrating advanced capabilities. The standardization for ROS-Industrial interfaces to manufacturing automation hardware results in decreased

integration costs by easing communications setup and by allowing software modules to be developed generically with broad applicability.

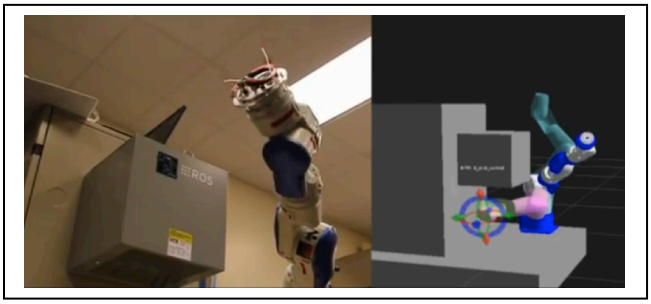


Figure 1. Screenshot of interactive motion planning environment (right) and physical robot hardware (left). The ROS-Industrial software plans an appropriate path in order to avoid collisions with nearby objects.

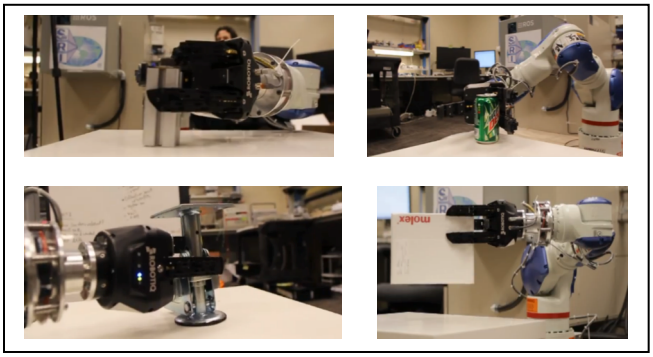


Figure 2. Images show autonomous grasping of multiple objects types.

V. Lessons Learned

A. Hardware Interfaces

While the integration of ROS within industrial applications has many benefits, it does not come without challenges. There are significant difficulties in adapting existing interfaces with ROS standard messages. Documentation for seldom used communications channels is often vague. There are always nuances in the way each vendor implements similar modes of operation. Streaming motion can be quite different between vendors. Some controllers require a buffer in order to achieve smooth motion, others do not. Vendor cooperation and support, as we have received to date, is key to creating a robust and reliable interface.

B. Reliability

One of the major hurdles for the ROS-Industrial program is achieving the required level of reliability for industrial software. While this is mostly a perception issue, rather than question of ROS' actual reliability, it still is of concern. The best way to address this issue is to extend the already existing quality assurance (QA) processes in ROS. Existing QA processes include software reviews and automated testing which is similar to the level of testing for industrial software. The ROS-Industrial program aims to address this concern by identifying those parts of ROS that meet the QA level required for industrial software robustness and reliability.

C. Complexity

The ROS-Industrial software is relatively complex when compared to existing commercial robot programming solutions. Software complexity is unavoidable when solving complex problems. However, the ROS-Industrial program is developing tools and programming interfaces, which simplify the programming process.

VI. Conclusion

In conclusion, the ROS-Industrial program aims to bring advanced ROS capabilities to industrial applications. Autonomous pick and place capability, a capability not commercially available today, has already been demonstrated. The vision for ROS-Industrial is to create a base library from which developers create industrial "apps". Such "apps" would be intuitive, flexible, and simple to deploy. Example apps might include paint application, bin picking, or mechanical assembly. If successful, ROS-Industrial will broaden the industrial automation market, while at the same time decreasing integration costs. The ROS-Industrial program will require support from both the open source and commercial communities. To that end, a ROS-Industrial Consortium is being developed to support and fund the continued development of ROS-Industrial.

Acknowledgment

The authors would like to acknowledge the support of their collaborators on the ROS-Industrial program, Yaskawa Motoman, Willow Garage, and Southwest Research Institute.

References

[1] G. Biggs and B. MacDonald, "A Survey of Robot Programming Systems," In Proc. of Australasian Conference on Robotics and Automation (CSIRO), Dec. 2003.

[2] ROS: an open-source Robot Operating System Quigley, Morgan., Conley, Ken., Gerkey, Brian P.., Faust, Josh., Foote, Tully., Leibs, Jeremy., Wheeler, Rob., and Ng, Andrew Y. ICRA Workshop on Open Source Software, (2009)

[3] 3D is here: Point Cloud Library (PCL) Rusu, Radu Bogdan., and Cousins, Steve International Conference on Robotics and Automation, 2011, Shanghai, China, (2011)

[4] Towards Autonomous Robotic Butlers: Lessons Learned with the PR2 Bohren, Jonathan., Rusu, Radu Bogdan., Jones, Gil E.., Marder-Eppstein, Eitan., Pantofaru, Caroline., Wise, Melonee., Mosenlechner, Lorenz., Meeussen, Wim., and Holzer, Stefan ICRA, 05/2011, Shanghai, China, (2011)

[5] "Robots using ROS." Internet: http://ros.org/wiki/Robots, Mar. 19, 2012 [Mar. 20, 2012].

[6] ROS Industrial" Internet: http://youtu.be/qd76wAywZos, Jan. 6, 2012 [Mar 21, 2012]

[7] Perception, Planning, and Execution for Mobile Manipulation in Unstructured Environments Chitta, Sachin., Jones, Edward Gil., Ciocarlie, Matei., and Hsiao, Kaijen IEEE Robotics and Automation Magazine, Special Issue on Mobile Manipulation, (In Press)

[8] "ROS Industrial Software Application" Internet: http://www.youtube.com/watch?v=_WG-45cZSUQ, Feb. 27, 2012 [Mar 21, 2012]

Using Human Gestures and Generic Skills to Instruct a Mobile Robot Arm in a Feeder Filling Scenario

Carsten Høilund, Mikkel Rath Pedersen, and Volker Krüger
Department of Mechanical and Manufacturing Engineering
Aalborg University, Copenhagen, Denmark
{*ch, mrp, vok*}*@m-tech.aau.dk*

***Abstract*— Mobile robots that have the ability to cooperate with humans are able to provide new possibilities to manufacturing industries. In this paper, we discuss the use of the mobile robot for a feeding scenario where a human operator specifies the parts and the feeders through pointing gestures. The system is partially built using generic robotic skills. Through extensive experiments, we evaluate different aspects of the system.**

I. Introduction

One very important element preventing industrial application of mobile robot arms is the lack of basic human–robot communication skills which has the effect that reprogramming the mobile arm is very resource demanding. It cannot be made by the operator on the shop floor; instead it involves various domain experts. One aspect of this paper is to to enhance our existing mobile robot arm [4] (see Fig. 2) that has already been tested at the shop-floor of the international company Grundfos [3] with the ability to interpret human actions and complex communicative gestures so that the mobile robot arm can be easily controlled through human gestures like *Take this object and place it there*. In the scenario considered here the robot will be used for automatically feeding production machines, see Figure 1. In a typical example situation a human advises the mobile robot arm to follow him. When the human reaches the work site he/she can point at a set of small load carriers (SLCs) as to give the command *take these*. The robot is able to interpret the pointing action and is able to identify where the human was pointing to, i.e., which SLCs the human was talking about. Then, the human may walk to a different location and point again at a place with the command *place here* in order to identify for the robot where it should place the SLCs. In terms of the feeding task, the command *empty in here* would cause the robot to empty the SLCs into the specified feeders. After these instructions, the robot is able to execute the commands autonomously. When the robot receives the signal *feeder nearly empty* either from a sensor in the feeder, from the production control system or from a human it will fetch one of the specified SLCs, move it to the specified machine and empty it into the feeder. Another very important aspect in building robots that are able to work in a human environment is the use of *skills*. Skills are building blocks that contain the necessary functionalities for sensing the environment and for planning. Furthermore, while commonly used robot macros are based on 3D Cartesian coordinates (e.g. precise 3D location of SLCs), skills are object oriented: It is sufficient for the human to specify an SLC by pointing at it. The sensory capabilities of the robot will then be used to compute the corresponding 3D location. This makes the robot robust to, e.g., location variations of the SLCs. Skills should eventually allow ordinary shop-floor workers to program the robot by offering an abstract and high-level programing interface.

*This work was partially supported by GISA, FP7-ICT-231143 - ECHORD

Fig. 1. The left image shows a typical production line at Grundfos A/S with the feeders marked. The right image shows a typical feeder.

The contribution of this paper is:

1) the first implementation of a mobile robot arm for use in a feeding scenario,
2) the recognition of human gestures, and
3) the use of robot skills for programming.

II. Related work

Similar to papers that model human action using an abstraction hierarchy of action primitives, actions and activities [6], [1], we will denote the same type of hierarchy by using the terms *skills*, *tasks* and *missions*. A *skill* is the basic building block for the robot actions. Available skills are grasping, driving, lifting, etc. *Skills* are also often called *action primitives* or *movement primitives*. Sequences of skills are considered a *task* (or *action*). To keep the feeders of the production line filled at all time is a mission (or activity). Most skills have parameters. E.g., the grasping-skill has a parameter with the information what to grasp. Modeling human actions for human-robot interfacing is an active research area, see [6], [7] for recent reviews.

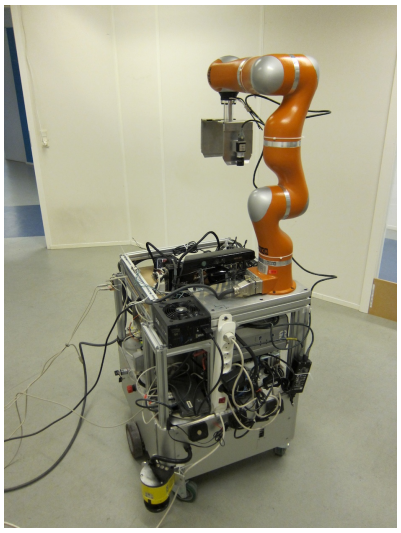


Fig. 2. This figure shows our mobile platform with the light-weight robot (LWR) arm and the Kinect camera. The LWR is currently parked in home position.

Human motion capture for action recognition is an open problem [6]. In robotics research a first focus was on the extraction of the task knowledge by observing and analyzing the changes in the environment caused by a human performing an assembly task [5], [8]. Consequently, it is ongoing research, to identify action units (action primitives, skills, etc.) as atomic entities and to use a grammatical abstraction for recognizing and modeling actions on a robot [8].

III. Mobile Robot Arm for Feeding Production Lines

We are using an off-the-shelf Kinect RGBD camera for sensing and a KUKA Light-Weight Robot (LWR) arm as an actuator. The camera is mounted next to the LWR on a Neobotix MP-L655 mobile platform (see Fig. 2). The camera is 80 cm from ground level. The used Small Load Carriers (SLCs) are open plastic boxes of size 21x15x31 cm. Hand-eye calibration of the robot between the Kinect, the visual camera and the robot end-effector is completely automatic.

IV. Recognition of communicative gestures

In this system, we have three distinct gestures. One is the pointing gesture where we are interested in not just the gesture itself but also the pointing direction. The other two gestures are a “Follow me” gesture and a “Stop following me” gesture, starting and stopping the program that would move the robot to keep the human operator within the field of view.

For human 3D tracking we use the OpenNI skeleton tracking software that is readily available in ROS. With the skeleton data available, we determine if the current state of the human joints indicate a certain gesture. For recognizing pointing gestures, we use our earlier work [9], [7].

For the “follow me” and “stop following me” gestures we require the operator to keep his right or left hand, respectively, in front of his chest for 4 frames. This approach has the advantage that it does not requiring any training of the system and it has a low computational complexity thus incurring very little run-time overhead.

V. Using robot skills

Skills are the foundation of task-level programming and provide the building blocks for the robot to complete the task. Which skills are available to a robot depends on its hardware and its sensors. Skills are effectuating a change in a set of state variables, describing the knowledge the robot has of its surroundings. State variables can be either measured with vanishing uncertainty by dedicated sensors, e.g, by those that are built into the manufacturing systems, or by sensors on the robot, such as vision, torque or tactile sensors.

One core property and main justification for using skills is their object-centeredness. Classic robot functions are usually based on 3D coordinates, e.g. a `pick up` function requires the object to be at an a-priori defined 3D location. Skills, on the other hand, are not applied on 3D locations but on objects, i.e. `pick up <object>`. In order to instantiate e.g. the `pick up` skill on `object`, the robot will use a sensing device such as a camera or a range scanner to first detect and then localize the object. Once the 3D location is available, the robot is principally able to execute the classic function for picking up the object.

A second core property of a skill is self-inspection; each skill needs pre- and postconditions to ensure and verify a correct functioning: Before the robot can execute a skill, all preconditions need to be fulfilled, e.g. reachability of the object is a precondition of the `pick up <object>` skill. If the object is not reachable, the skill cannot be executed. Instead, the robot will have to call the `move to <location>` skill that will bring the robot within reach of the specified object. A check of the postconditions will verify if the expected outcome of the skill was satisfactorily met, i.e. that the executed skill was successful. Thus, the pre- and postconditions are effectively a query on the state variables, that evaluates to true or false. This also necessitates that the skill needs knowledge of the expected outcome, a prediction of the change in the world state.

Robot skills have two very distinct features; *execution* and *inspection*, each requiring a different form of object interaction. Thus, a robot skill is expected to modify the state of the real world and concurrently update the systems state variables. A model of a robot skill is shown in Fig. 3. Queries on the state variables and input parameters (which are provided at task-level programming time) serves as a means of testing if the preconditions of the skill execution are met, either by prior knowledge or ad hoc inspection. If the preconditions are satisfied, the skill is executed based on the parameters and the state variables. Parameters are thus stored in the task description and are the objects, the skill is performed on, e.g. `<red box>` for the `locate` or `pick up` skill or `<warehouse>` for the `move to` skill.

The postconditions are two-part in relation to the skill; prediction and evaluation. The prediction specifies formally what the expected effect of executing the skill is, and can

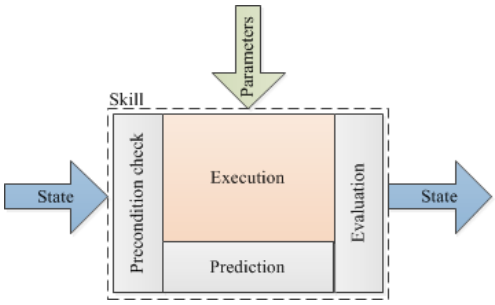


Fig. 3. Skill Model

Fig. 4. This figure shows the experimental setup in our lab.

thus be used to select an appropriate skill (or set of skills) for achieving a desired goal state. The evaluation checks that the state variables after execution is within an expected range and updates the state variables to reflect the actual state after the skill execution. By implementing skills in the manner described above, it will eventually facilitate a number of beneficial aspects, e.g.

- better overview of the programming phase for all actors (shop floor workers, robotic engineers, system integrators, etc.),
- task-level programming, by selecting a sequence of skills and their parameters, and
- planning a sequence of skills to achieve a certain goal state.

VI. Experiments

A. Gesture recognition

We verified the gestures could be successfully recognized, which is sufficient for the "follow me" and "stop following me" gestures. However, the pointing gesture is a different matter, since it is not simply a matter of whether or not it could be recognized, but also how accurate the extracted pointing direction is.

To determine how well the recognition of the pointing direction works and how good untrained humans are in communicating to a robot via pointing we carried out a set of experiments:

- 10 volunteers participated in the evaluation. They were male and female evenly distributed with ages 22 - 50, and 7 volunteers were without technical background and had never worked with robots before. Each volunteer pointed with their right hand at each box 3 times.
- We have tested 5 different locations within the field of view of the camera: 2 m, 3 m, and 4 m from the operator to the camera when directly in front of the camera, and in the left and right side of the field of view of the camera when at the 3 m mark.

The human operator chooses one of three boxes by pointing at the box. The boxes are placed adjacently on a table 70 cm from the front of the robot, ordered from left to right as Blue, Red, and Green (from the point of view of the human volunteer). With the position of the boxes stored as the center of the barcode, the effective distance between the points in space the human operator must distinguish between using pointing gestures is 21 cm. The operator is standing behind the table within the field of view of the Kinect camera. The goal is to have the robot determine which box the operator is pointing at.

Fig. 4 shows the experimental setup in our lab for testing the pointing accuracy.

1) Results: Our system has so far only been tested in our lab, but the real test at Grundfos is upcoming. The robot itself has already been tested [4], and given that the Kinect has proven to be robust, we do not expect the industrial test to be of any difficulties. Furthermore, we have tried to keep realistic variations within the demo, e.g., by letting the robot move and thus excert the usual uncertainty in localization with respect to the environment. Thus, key issues for testing were the use of the gestures and the use of the skills.

The overall percentage of correctly recognized pointing directions at distances of 2 m and 3 m are 95.3% and 94.4%, respectively. The recognition results at 4 m were still 79.9%, see [2] for details. The increase of error at 4 m is due to the fact that the Kinect is able to reliably track only up to 3.6 m. After that, tracking reliability indeed strongly degrades. Another reason for tracking failures was that the lower part of the human body was occluded and that for the 2 m distance, even the hands were sometimes occluded by the SLCs which resulted in lost tracks.

A cause for error is that the position of the box is presently stored as a single point in space, specifically the center of the front of the box as seen by the camera. Since participants could point at the box as they wanted, and thus did not necessarily point at the exact location of the barcode, the pointing ray when pointing from the left or right side of the field of view of the camera could be closer to a point representing one of the other two boxes.

We also investigated the influence on the pointing recognition results when the person was standing in the leftmost and rightmost location within the field of view of the camera, where the obtained accuray was 82.86 % and 97.22 %, respectively.

One might argue that humans with robotics experience would potentially give better pointing directions to the robot. Thus, we have compared the recognition of the pointing

results of the experienced users with the unexperienced ones. The results clearly state that the quality of the pointing recognition is independent trom the user experience and training.

B. Skill implementation

In this experiment a full implementation of skills, as described in section V, has not been carried out, since this preliminary experiment is merely a proof-of-concept. The key aspects of skills in the experiment includes *a)* a simple set of state variables, *b)* object-centeredness and *c)* parameter-based execution, and the self-inspection is lacking.

For the bin-filling scenario in this experiment, the state variables are simply chosen to be the states of the gripper and the box of interest. Thus, the `gripper` state variable can assume the values {`full, empty`}, to signify if the robot is carrying a box in the gripper, and the `box` state variable the same values, to signify whether or not the carried box contains parts. The settings of the two state variables are changed upon the execution of a skill, e.g. the execution of the `pick up <box>` skill sets the states `gripper: full` and `box: full`. The feeder state is not included in this experiment, since we are not experimenting on the mission level in this setup.

Parameter-based execution and object-centeredness are intertwined in this experiment, since the state variables dictate which *type* of object should be located, and the parameter specify which *particular* object should be the focus of the skill. Based on the settings of the state variables, and given a pointing gesture, the system looks for all objects of a specific type in the scene, and locates the one closest to the direction of the pointing motion. E.g. if the robot is carrying an empty box in the gripper, and observes a pointing gesture, the system looks for an object of the type *shelf* where it can put down the empty box.

When the parameter is extracted through the method mentioned above, the location and orientation of the object is determined, using the QR code on the object. This location is then used in the skill execution. The execution phase can thus be divided into a motion through some hardwired locations, regardless of the parameter, and a motion specified by the object location, e.g. the location of a feeder specifies where the robot should perform the actual unloading motion. The skill execution is thus the lowest abstraction level in this experiment. Upon execution of the skill, the state variables are updated, and the system returns to a waiting state.

A video of the feeding demo in our lab can be seen at `http://www.youtube.com/watch?v=tusLjLp1r64`.

VII. Conclusions

The mobile robot platform has been tested on the shopfloor at Grundfos [3] without the use of human gesture recognition, whereas the human gesture recognition has only been tested in our lab, and remains to be tested on the real shop floor. However, given the success and the robustness of the Kinect camera, we do not expect any major difficulties. All software components are implemented in ROS.

The robot system has been partially implemented using skills. The experimental results already reflect that the use of skills will define a paradigmatic shift in robot programming: Instead of using the classic linear programing, the use of skills offers task-level programing [5], [8], which is a completely different programming paradigm. It is ongoing work to develop a planner and the corresponding processing infrastructure to support non-linear, task-level programing and STRIPS-like planning.

References

[1] A. Bobick and V. Krüger. *Visual Analysis of Humans, T. Moeslund and A. Hilton and V. Krüger and L.Segal (Eds.)*, chapter On Human Action, pages 279–288. Springer, 2011.

[2] Thomas Moeslund Carsten Hoilund and Volker Krueger. Communicating with robots through human pointing gestures. In *Proceedings Robotics, Science and Systems*, 2012.

[3] www.grundfos.com.

[4] Mads Hvilshøj, Simon Bøgh, Oluf Skov Nielsen, and Ole Madsen. Multiple part feeding - real-world application for mobile manipulators. *Assembly Automation*, 2011.

[5] K. Ikeuchi and T. Suchiro. Towards an assembly plan from observation. i. assembly task recognition using face-contact relations (polyhedral objects). pages 2171–2177. IEEE Comput. Soc. Press, 1994.

[6] V. Krueger, D. Kragic, A. Ude, and C. Geib. The meaning of action: A review on action recognition and mapping. *Int. Journal on Advanced Robotics, Special issue on Imitative Robotics, T. Inamura and G. Metta (eds.)*, 21(13):1473–1501, 2007.

[7] V. Krueger, Sanmohan, D. Herzog, A. Ude, and D. Kragic. Learning actions from observations. *IEEE Robotics and Automation Magazine*, 17(2):30–43, 2010.

[8] Y. Kuniyoshi, M. Inaba, and H. Inoue. Learning by watching: extracting reusable task knowledge from visual observation of human performance. *IEEE Transactions on Robotics and Automation*, 10(6):799–822, December 1994.

[9] T.B. Moeslund, M. Strring, and E. Granum. A natural interface to a virtual environment through computer vision-estimated pointing gestures. In Ipke Wachsmuth and Timo Sowa, editors, *Gesture and Sign Language in Human-Computer Interaction*, number 2298 in LNAI. Springer, 2001.