proceedings

MDPI

Proceedings

Real-Time Posture Control for a Robotic Manipulator Using Natural Human-Computer Interaction †

Guillaume Plouffe [1,*], Pierre Payeur [2] and Ana-Maria Cretu [2]

[1] School of Electrical Engineering and Computer Science, University of Ottawa, Canada; gplou010@uottawa.ca
[2] School of Electrical Engineering and Computer Science, University of Ottawa, Canada; ppayeur@uottawa.ca
[3] Department of Systems and Computer Engineering, Carleton University, Canada; acretu@sce.carleton.ca
* Correspondence: gplou010@uottawa.ca; Tel.: +01-613-562-5800
† Presented at the 5th International Electronic Conference on Sensors and Applications, 15–30 November 2018; Available online: https://sciforum.net/conference/ecsa-5.

Received: date; Accepted: date; Published: date

Abstract: In this paper, we propose a vision-based recognition approach to control the posture of a 3 DOF robotic arm using static and dynamic human hand gestures. Two different methods are investigated to intuitively control a robotic arm posture in real-time using depth data collected by a Kinect sensor. In the first method, the user's right index fingertip position is mapped to compute the inverse kinematics on the robot. Using the FABRIK algorithm, the inverse kinematics solutions are displayed in a graphical interface. Using this interface and his left hand, the user can intuitively browse and select a desired robotic arm posture. In the second method, the user's left index position and direction are respectively used to determine the end-effector position and an attraction point position. The latter enables the control of the robotic arm posture. The performance of these real-time natural human control approaches are evaluated for precision and speed against static and dynamic obstacles.

Keywords: posture control; robot end-effector; natural human-computer interaction; gesture recognition

1. Introduction

The ability to control complex systems with minimum cognitive strain, physical constraints and physical effort has been the focus of research for many decades in the field of robotics. While controlling a multiple joint robotic arm is possible without constraints in time and space, it becomes extremely difficult when trying to quickly reach a target while avoiding obstacles (in motion or static). A human user can control the movement of his arm and finger joints with almost no effort within a constrained space. The human capability to interpret arm and finger movement provides an interesting asset in exploring new approaches for the movement control of a robotic arm. While the number of joints and constraints associated to a human arm might not correspond to those of robotic arms, it is nevertheless possible to interpret the human postures or gestures to intuitively provide the intended joint angle commands to the robotic arm.

In the context of master-slave robotic systems involving the human arm in the master role; many solutions exist that differ in their methods and sources of data collection. Haptic devices [1-3] reproducing the robotic arm motion are used for the robot to imitate the haptic device posture moved by hand by an operator. Alternatively, an exoskeleton [4-5] is worn around the user's arm to control the robotic arm joints, or a digital glove [6] is used to control the end-effector position. In [7], a marker system is mounted on the master (human) to map directly his joint movement to the ones of the robotic arm. To control the robot, the FABRIK inverse kinematics (IK) method [8] can be used to avoid singularities and speed up calculation, while achieving good performance in the tracking of the target

by the end-effector. In [9-11], a Kinect-based solution controls the end-effector, but does not consider the robotic arm posture. A markerless approach offers simplicity, low cost to build and less physically cumbersome user experience. These reasons motivated to use a markerless strategy in our initial work, which consisted of recognizing static and dynamic human hand gesture based on depth data collected by a Kinect sensor [12]. Additionally, mimicking the exact movement of the human arm as proposed in [7] might be optimal in term of intuitive control when the robot has a similar configuration (DOF, type of joints, etc) and constraints (joints angle limits) compared to the human arm, but can become cumbersome or plainly impossible to apply otherwise. The work presented in this paper is an extension of our initial work in [12], which makes use of the recognized gestures captured by a Kinect sensor to naturally control a robotic arm.

One of the main problems in remotely controlling a robotic arm is the avoidance of obstacles, either static or dynamic. The solutions proposed in [13, 14] target pre-known static obstacles in the proximity of a fixed robotic arm, that impose a space of possible movements for the robotic arm. In the work of Ju et al. [15], a rapidly exploring tree method is used to calculate a trajectory path free of obstacles to reach a designated target. This approach encompasses time-consuming calculations and thus can be difficult to apply in changing environments. In [16], a multi-tasking approach involves human motion imitation for obstacle avoidance moving the end-effector along a desired trajectory. It was later improved [7] by replacing human motion imitation by human gesture imitation. In teleoperation, approaches were investigated to control the robot using torque sensors [17] with impedance, admittance and stiffness control. These methods are more complex, and prone to instability and inaccuracy when compared to a vision-based system with inverse kinematics.

This paper investigates an alternative way for resolving the obstacle avoidance problem in conjunction with a given task sent to a remote robotic arm. The proposed approach uses teleoperation based on a natural gesture recognition system that can intuitively and naturally allow the user to control both the end-effector and the overall posture of a given robotic arm in real-time. To avoid singularities and to increase the speed of inverse kinematics calculation in the search of possible postures, the FABRIK method is exploited. A first proposed solution involves a graphical interface containing a visual display of the robotic arm with all possible postures. The user can browse and select the desired new posture in real-time using a dedicated hand, while the other hand positions the end-effector. A second proposed solution tracks a fingertip for providing the position of the end-effector, where the new desired posture is indicated by the fingertip direction. In this case, the direction will correspond to the way the robotic arm joint will bend.

2. Teleoperation with dedicated hand for posture control – Browse and select

In this method, the user relies on a visual display indicating in real-time the available postures (possible inverse kinematics solutions) in the vicinity of the current robotic arm configuration. The user can browse the available postures using his left hand index and select one of them by making a V sign with two fingers to the Kinect sensor. This approach forces to user to constantly shift his attention from the graphical interface displaying the posture to the actual robotic arm to evaluate his next decision. It is therefore more appropriate for tasks where posture accuracy is very important, over the speed of execution. In Figure 1, the browse (Figure 1(**a**)) and select (Figure 1(**b**)) process is presented with N=4 possible postures in a sequence where pink lines represent (non-selected) possible postures of the arm, the red line is the currently selected posture, and the blue line is the current posture.

For our experimentation, we used a simulation of the AL5D robotic arm from Lynxmotion. The AL5D uses a joint at the base that rotates on a perpendicular plane relative to the other revolute joints. This joint is excluded from our simulation, thus only 3 DOF are considered. Our algorithm finds a number N of possible posture solutions, each capable of reaching the same target end-effector position. This algorithm works only for the neighboring set or subset of joints that revolute along the same plane (2D). These postures are displayed in a graphical interface allowing the user to browse and select them. To find different solutions, we need to input different initial arm configurations to the FABRIK inverse kinematics algorithm. The FABRIK algorithm starts by geometrically positioning

the end-effector (joint i) at the target location, then moving joint i-1 at a distance equal to the length of the link between joint i and i-1 from the target on a vector crossing the new position of joint i and the old position of joint i-1. It then continues down to the base joint of the robotic arm. When reaching the base joint, this joint will be positioned at a new location following the algorithm logic. However, since the base joint is fixed and cannot be moved, it will be geometrically repositioned at its old location. The algorithm then corrects each joint position accordingly following the same logic it used going down the arm. This process iterates up and down the arm until the position of both the end-effector and the base reach their targeted position.

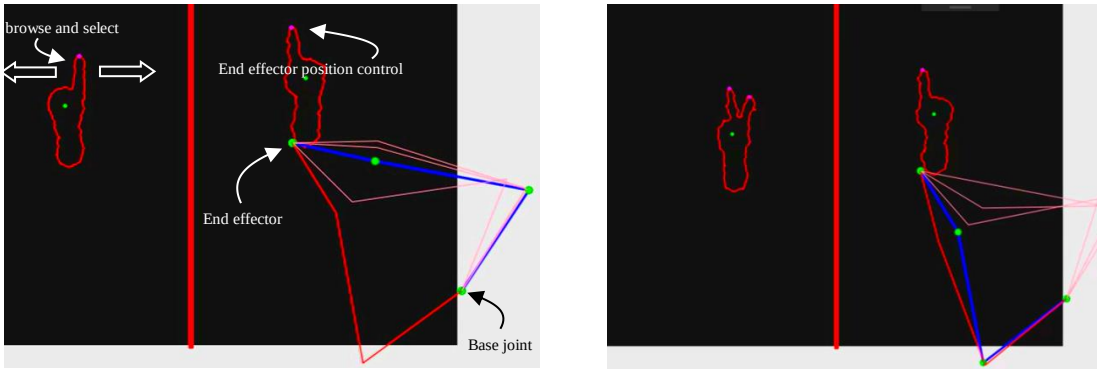


Figure 1 (a) The next posture selected (in red) with respect to the current robot arm posture (blue) can be changed among other possible postures (in pink) by moving the left hand finger towards the left or right direction; (b) The currently selected configuration becomes the new posture (change from red to blue) and a new set of possible postures is generated when the two finger sign is detected on the left hand.

3. Teleoperation with posture control using a single hand – Attraction point

The second approach leverages the ability of the gesture recognition system to detect the angle between the axis of the finger passing through the fingertip and the reference axis in the Kinect view plane. This angle is directly mapped to the angle between the vector going from the middle point between the end-effector and the base joint to the attraction point (pink line in Figure 2) and the vector going from the base joint to the end-effector as illustrated in Figure 2.

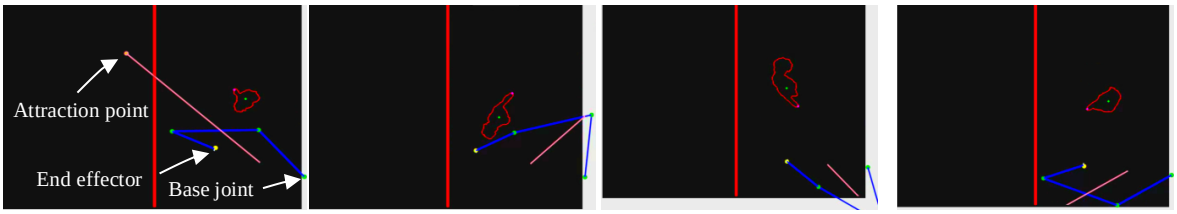


Figure 2 (**a**)-(**d**) Demonstration of the gesture control using an attraction point controlled by a user's finger direction. The 3 DOF robotic arm is indicated by the blue line and the pink line refers to the directional vector between the attraction point and the middle point between the end-effector and the base joint.

An obvious benefit of this approach is that only one hand is needed by the user to control both the end-effector position and the arm posture, leaving the second hand free to potentially control another robotic arm. The approach can be seen as complementary to the browse and select solution presented in section 2, as it is less precise but more intuitive and faster to use. Indeed, this approach can only control a limited number of the possible postures because only one attraction point is under the finger direction control. In this method, the same finger used to position the end-effector also

controls the selection of the robotic arm posture by measuring the finger's relative angle in the X-Y plane of the Kinect field of view. The resulting angle is then used to position an attraction point around the robotic arm. To change the posture, the attraction point, G, draws each moveable joint, j, (i.e. all joints except the base joint and the end-effector) closer in its direction. The algorithm below was designed to find the posture where the sum of the distances of each of its moveable joints relative to the attraction point is minimal. In the algorithm each calculated distances (Distance(G, j)) is raised to the power of 3 to avoid postures with very distant joints. The attraction point is positioned on the perimeter of a circle of radius equal to the maximum reach of the robotic arm (L_{MAX}) and centered half way between the end-effector and the base joint (L/2) as illustrated in Figure 3. No matter the arm configuration (i.e. number of joints, length of links) and its current posture, this solution makes sure the attraction point is always outside the circle containing all the joints.

Algorithm for searching for closest posture to equilibrium based on the attraction point position
Input: $\mathbf{L_P}$ = List of possible arm postures **G** = Attraction point position Output: **A** = Possible posture that is the closest to equilibrium
1. For each posture *i* in $\mathbf{L_P}$: For each moveable joint *j* in *i*: DistanceTotal += (Distance(**G**, *j*))^3 if(DistanceTotal < minDistanceTotal) minDistanceTotal = DistanceTotal **A** = *i*; 2. return **A**

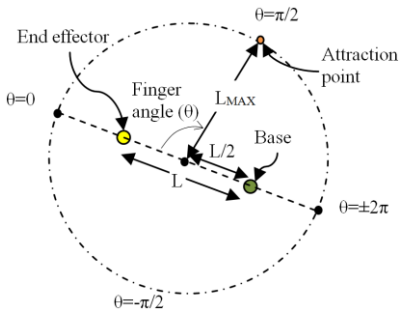
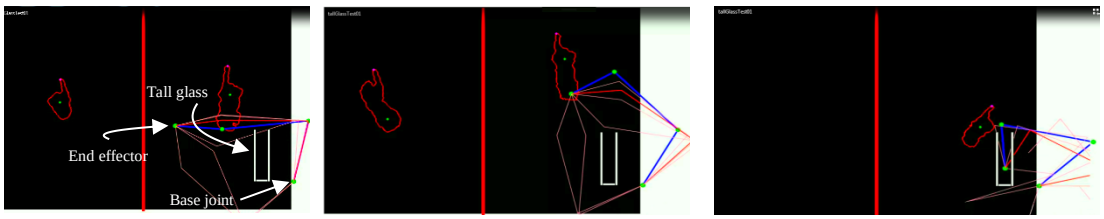


Figure 3. Position of the attraction point relative to the arm and the finger angle (θ).

4. Experimental results

4.1. Static obstacle avoidance – The tall glass test

In the first experiment, the ability of the browse and select posture control described in section 2 is tested to reach the bottom of a tall glass without touching the sides. This is an almost impossible task when relying exclusively on moving the end-effector without controlling the posture. For this experiment, the virtual glass is 26 pixels wide by 124 pixels tall (equivalent to 2.6 x 12.4 cm) and it is shown in white in Figure 4. Each test begins by choosing the first posture that goes over the glass and a timer starts to monitor the efficiency of the procedure (Figure 4(**a**)). The first posture selected is rarely an adequate posture to allow reach the bottom of the glass with no arm link touching the sides. Hence, posture selection choice must be refined with another possible posture near the current robotic arm posture, to achieve a posture allowing the user to move the end-effector toward the bottom of the glass by only controlling the end-effector (Figure 4(**b**)). The user can control only the end effector by removing his left hand from the Kinect recognition field of view. The process gets easier and faster with practice. We stop the time when the end-effector reaches the bottom of the virtual glass (Figure 4(**c**)).



(**a**) (**b**) (**c**)

Figure 4 (**a**) First posture selected over the glass; timer starts; (**b**) Browse and select to get to the posture capable of reaching the bottom of the virtual glass, and (**c**) End-effector reaches the bottom of the virtual glass. The timer stops.

For the tall glass experiment, 15 tests were realized where each test was successful in reaching the bottom of the virtual glass without touching the sides of the glass. The shortest test took 10 seconds while the longest took 67 seconds. The average time was 29.93 seconds to complete each test.

4.2 Dynamic obstacle avoidance – Moving block

In the second experiment, a virtual block is repeatedly moving vertically and horizontally in a configuration inspired from the experiment realized in [7], where the authors tested the performance of their solution with moving obstacles while attempting to execute a simple task like drawing a line and a height figure. For this experiment, the speed of displacement for the obstacle was set to 1 pixel/100ms, which is equivalent to 1cm/s when mapped to real world units. In this scenario, the goal is to pick an object with the end-effector in recipient A, to drop it in recipient B and to return to recipient A without any part of the robotic arm touching the moving obstacle (block) as illustrated in Figure 5. The obstacle (moving block) starts at the top of the vertical path, moves down the entire length of the vertical path, goes up a little then goes left and right on the horizontal path before returning up to its initial position. This movement is then repeated over and over.

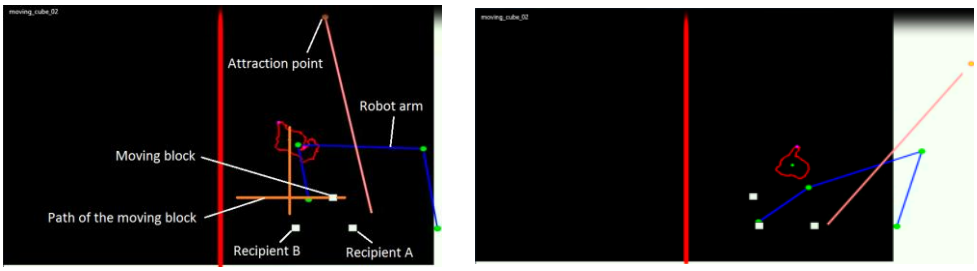


Figure 5 (**a**) The end-effector tries to avoid the moving obstacle while transporting an object; (**b**) the end-effector successfully drops an object in recipient B.

For this experiment, a maximum number of objects was picked and dropped by the user in a limited time. The time taken between each pick was recorded. This resulted in a total of 49 successful pick and drops with no collision between the arm and the obstacle. On average, it took 10.92 seconds between picks, with the longest being 23 seconds while the shortest 5 seconds. Longer times were observed when the arm had to go over the obstacle, moving between the recipients, to drop the object. As a comparison, we also performed this test using the first proposed method in section 2, and with a simple end-effector tracking technique with no posture control. But in both cases the obstacle was consistently hitting the robotic arm. The obstacle was moving too fast for the former and was continuously obstructing the way when using the latter method.

5. Conclusion

Two complementary vision-based hand gesture recognition approaches were proposed to control the posture a 3 DOF robotic arm in a simulation environment using the FABRIK algorithm for IK calculation. The first approach relies on a dedicated hand to browse and select possible postures of a robot arm. It revealed more appropriate for tasks requiring precision over speed. The second posture control approach addresses problems related to dynamic environments were obstacle position can change in time. Even if this approach has a limited spectrum of postures for control, we can still leverage its speed as presented in the moving block experiment. As we have seen, in such a dynamic environment, the first proposed approach or a simple end-effector control approach would have great difficulty in avoiding the obstacle. Both proposed solutions are applicable to different robotic arm configurations and applying them on real robotic arms will be part of future research.

Author Contributions: G. Plouffe developed the concept, implemented the framework and performed experiments; all authors jointly wrote the paper; P. Payeur and A.-M. Cretu developed the concept and supervised the research.

Funding: The authors acknowledge support from Natural Sciences and Engineering Research Council of Canada (NSERC).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. C. Pan, X. Liu and W. Jiang, "Design and Synchronization Control of Heterogeneous Robotic Teleoperation System," in *2017 Chinese Automation Congress (CAC)*, Oct. 2017, pp. 406-410.
2. T. Ma, C. Fu, H. Feng and Y. Lv, "A LESS Robotic Arm Control System Based on Visual Feedback," in *IEEE International Conference on Information and Automation*, August 2015, pp. 2042-2047.
3. L. Barbeiri, F. Bruno, A. Gallo, M. Muzzupappa and M.L. Russo, "Design, prototyping and testing of a modular small-sized underwater robotic arm controlled through a Master-Slave approach," *Ocean Engineering*, Vol 158, June 2018, pp. 253-262.
4. A. Chai and E. Lim, "Conceptual Design of Master-Slave Exoskeleton for Motion Prediction and Control," in *2015 International Conference on Advanced Robotics and Intelligent Systems (ARIS)*, May 2015, pp. 1-4.
5. M. Fuad, "Skeleton based gesture to control manipulator," in *2015 International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation (ICAMIMIA)*, Oct. 2015, pp. 96-99.
6. M. R. Huertas, J. R. M. Romero and H. A. M. Venegas, "A Robotic Arm Telemanipulated through a Digital Glove," in *Electronics, Robotics and Automotive Mechanics Conference (CERMA 2007),* Sept. 2007, pp. 470-475.
7. H.-I. Lin, and X.-A. Nguyen, "A manipulative instrument with simultaneous gesture and end-effector trajectory planning and controlling," in *Review of Scientific Instruments*, Vol 88, no 5, May 2017, 055107.
8. M. A. Y. Abdallah, M. S. Baziyed, R. Fareh and T. Rabie, "Tracking Control for Robotic Manipulator Based on FABRIK Algorithm," in *Advances in Science and Engineering Technology Int. Conf.*, UAE, 2018, pp. 1-5.
9. H. Wu, M. Su, S. Chen, Y. Guan, H. Zhang and G. Liu, "Kinect-based robotic manipulation: From human hand to end-effector," in *IEEE Conf. on Industrial Electronics and Applications (ICIEA)*, June 2015, pp. 806-811.
10. I. Benabdallah, Y. Bouteraa, R. Boucetta and C. Rekik, "Kinect-based Computed Torque Control for lynxmotion robotic arm," in *Int. Conf. on Modelling, Identification and Control (ICMIC)*, Dec. 2015, pp. 1-6.
11. F. Marić, I. Jurin, I. Marković, Z. Kalafatić and I. Petrović, "Robot arm teleoperation via RGBD sensor palm tracking," in *Int. Conv. on Inf. and Comm. Technology, Electronics and Microelectronics*, 2016, pp. 1093-1098.
12. G. Plouffe and A.-M. Cretu, "Static and Dynamic Hand Gesture Recognition in Depth Data Using Dynamic Time Warping," *IEEE Trans. Instrum. and Measurement*, Vol. 65, No. 2, Feb. 2016, pp. 305-316.
13. M. Bensaoui, H.Chekireb and M.Tadjine, "Redundant robot manipulator control with obstacle avoidance using extended Jacobian method," in *Mediterranean Conf. Control and Automation*, June 2010, pp. 371-376.
14. K.-K. Lee, Y. Komoguchi and M. Buss, "Multiple Obstacles Avoidance for Kinematically Redundant Manipulators Using Jacobian Transpose Method," in SICE Annual Conference, Sept. 2007, pp. 1070-1076.
15. T. Ju, S. Liu, J. Yang and D. Sun, "Rapidly Exploring Random Tree Algorithm-Based Path Planning for Robot-Aided Optical Manipulation of Biological Cells," in *IEEE Trans. on Aut. Science & Eng.*, Vol. 11, Issue 3, July 2014, pp. 649-657.
16. A. A. Maciejewski and C. A. Kein, "Obstacle Avoidance for Kinematically Redundant Manipulators in Dynamically Varying Environments," *The Int. Journal of Robotics Research*, Vol 4, Issue 3, 1985, pp. 109-117.
17. R. C. Luo, B.-H. Shih and T.-W. Lin, "Real Time Human Motion Imitation of Anthropomorphic Dual Arm Robot Based on Cartesian Impedance Control," in *IEEE Int. Symp. Robot. & Sensors Environ. (ROSE)*, USA, Oct. 2013, pp. 25-30.