Gesture-Based Teleoperated Grasping for Educational Robotics

Alexander Koenig*[,1], Ferdinando Rodriguez y Baena[2] and Riccardo Secoli[3]

***Abstract*— We present an interactive robotic platform for teleoperated grasping as an educational tool. With this open-source robot application, we engage children and young adults with robotics and make computer science education more vivid. Our teleoperation method uses the Leap Motion optical gesture tracker to simultaneously control each of the four degrees-of-freedom (DOF) of a robotic hand and the six-DOF tool pose of a serial manipulator. A control algorithm is developed to relate the operator's palm pose to the manipulator's tool pose. The operator commands the robotic hand with relative finger movements of the thumb, index, and middle finger. We present preliminary results from a pick-and-place demonstration showcased at a public science fair held at Imperial College London.**

I. Introduction

The latest market trends show that robotics is a rapidly growing field. In the last decade, the number of operational industrial robots worldwide has almost tripled to 2.7 million in 2019 [1] and is estimated to reach 20 million by 2030 [2]. Consequently, the demand for robot service personnel and robotics researchers is forecast to rise. Conversely, a 2015 study by the European Union reports that only two-thirds of the respondents had a favorable view of robots, indicating a decline by six percent compared to their previous survey in 2012 [3]. New educational tools that engage children and young adults with robotics will help them in fostering a positive emotional response to robots and can inspire them to pursue a career in this domain to meet the ever-growing demand for roboticists.

Research shows that children and young adults, in particular, can benefit from new educational robotics tools. A recent study demonstrated that young adults are significantly more likely to engage with robots than elderly persons [4]. Moreover, the positive effect of educational robotics on children is widely acknowledged. Using robotics as an educational tool for pupils can spark interest in Science, Technology, Engineering, and Mathematics (STEM) subjects, promote collaboration among peers, improve social skills [5], [6] and improve children's measured spatial ability [7]. Studies also show that engaging students with robotics to teach computer science concepts is both valuable at the university level [8], [9], as well as at the elementary school level [10].

[1] Department of Informatics, Technical University of Munich
[2] Department of Mechanical Engineering, Imperial College London, Exhibition Road, SW72AZ London, UK
[3] The Hamlyn Centre for Robotic Surgery, Imperial College London, Exhibition Road, SW72AZ London, UK
email:r.secoli@imperial.ac.uk
* This author received funding from the European Union's Erasmus+ program

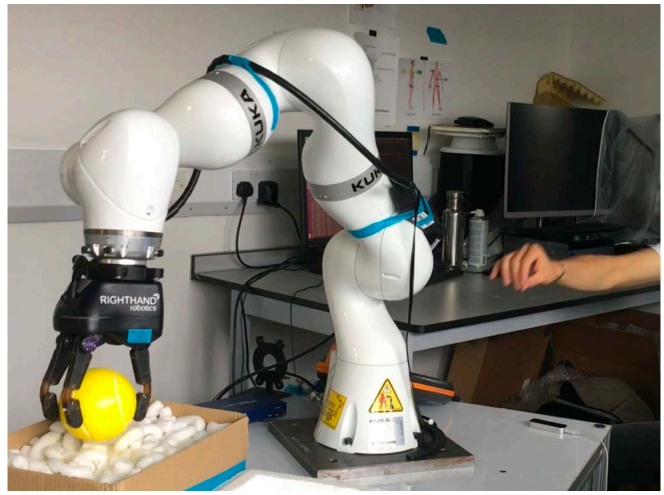

Fig. 1: Demonstration of interactive grasping task.

Several prior works presented robotic applications to engage children with robotics. Sullivan et al. [11] proposed Kibo, a mobile robotics toolkit designed for children between 4 and 7 years old. It engages children in exploring robotics concepts through tangible programming blocks and a modular robot architecture consisting of different actuators, wheels, lights, and sensors. Other works use similar mobile robot platforms to design robotics curricula engaging primary [12] and secondary school students [13], [14], [12], [15], [16] with STEM. Furthermore, humanoids robots, such as the Nao or Darwin, were used in educational programs to illustrate robotics concepts, such as computer vision or speech recognition [17].

Robotic arms and gripping devices are widely used tools in industry and thus play an important role in educational robotics. While robotic arms and grippers serve as teaching devices at the university level [18], [19], [20], they are far less prevalent in educational robotics for children. As demonstrated above, primary and secondary education mainly relies on simple mobile robot platforms and less on articulated robots. Platforms aimed at children are generally simpler and less interactive, at least in the context of real-world manipulation. However, it is through these very interactions that a deep and synoptic understanding is achieved in the young [21], [22].

Interaction is fundamental, and a master interface can play an essential role in the engagement and immersion of users [23]. The advent of modern motion tracking systems has sparked an increase in the use of human hand gestures for human-computer interaction [24]. We hypothesize that a gesture-based master interface for teleoperating a serial manipulator can enhance and naturalize interactions and thus provides better user engagement in the context of educational robotics.

Hand motion tracking interfaces, such as the Leap Motion (LM) [25], were previously used as a master interface for teleoperated robotic grasping. We found that the Leap Motion's potential as a natural user interface was often not fully exploited. Prior works presented Leap Motion control of three-fingered grippers and six-DOF serial robots in the context of elderly care [26], virtual-reality assisted teleoperation [27], [28], and simulated environments [29]. [26] rely on binary *grasp* and *release* commands for the gripper which the operator triggers with corresponding hand gestures. On the other hand, [27], [28], [29] linearly map the closure of the human hand to the simultaneous closure of all robotic fingers as a percentage $\in [0, 1]$. In these control frameworks [26], [27], [28], [29] it is not possible to actuate fingers individually, and therefore, much of the three-fingered robotic hand's dexterity is lost. More importantly, a lack of natural movement transfer from humans to robots is bound to impact both immersion and engagement negatively. Some related works present LM control of serial manipulators with one-DOF parallel-jaw grippers [30], [31], [32], which may provide a good solution for simple grasping tasks. However, such non-anthropomorphic grippers are less suitable in engaging users, as human-like characteristics in technology are an important factor for user interaction [33]. Furthermore, the presented works do not explicitly focus on system safety and usability, which is essential when targeting children and other non-technical users.

This work presents a case study for an educational software framework aimed at human interactive robotic grasping, formed by three components: a hand tracking device, a robotic three-fingered anthropomorphic gripper, and a serial robot. The framework was designed with a strong emphasis on safety and simplicity to make the tool usable for the robotics education of technical novices. Our framework for gesture-based teleoperation allows actuating all four DOF of the robotic gripper individually while simultaneously controlling the gripper's six-DOF pose, which provides a natural imitation of human grasping using three fingers.

II. System Overview

A. System Requirements

The primary system requirements for the robotic platform are *interactivity*, *safety*, and a *simple programming structure*. We create an *interactive* teleoperation experience by using the Leap Motion optical gesture tracker as the master interface. Gesture control provides an intuitive interface for the novices in our target group, especially compared to traditional control methods such as 2D joysticks or pre-programmed motion plans. To meet our high *safety* requirements, we combine the integrated hardware sensors of the robot with safety software modules. This combination creates safe and constrained workspaces suitable for deploying the robotic platform at public demonstrations. Lastly, we base our software framework on the Robot Operating System (ROS) [34] to achieve a *simple programming structure*. Our platform provides an excellent opportunity for students to get acquainted with this widely used library. Our packages are available as an open-source tool[1], and since our software is ROS-based, it is easy to modify and extend according to specific applications that can go beyond simple demonstrations.

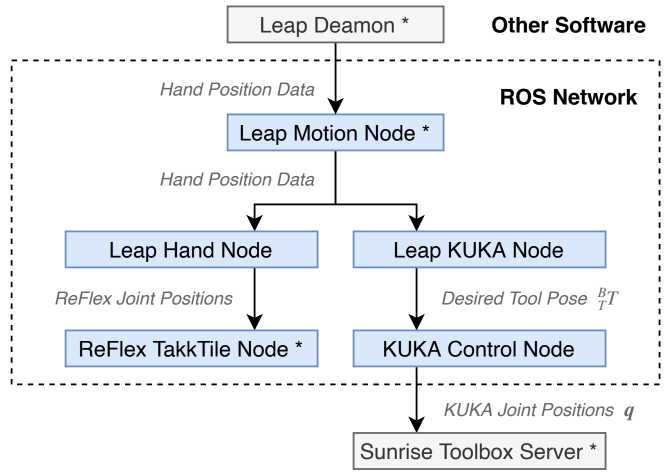

Fig. 2: ROS network overview (* is third party software).

B. Hardware Components

The robotic rig comprises three integrated hardware components: a hand tracking device, a serial robot, and a three-fingered robotic gripper. The hand tracking device is the Leap Motion, a markerless optical gesture tracker based on infrared stereo imaging [35]. The device provides geometric information of hands or tools recognized in its field of view (FOV). The serial robot used in this case study is the KUKA LBR Med 7 R800, a 7 DOF cobot with a 7kg payload [36] mounted on a table. The proposed software can work with other serial robots that may be more affordable than the KUKA LBR by substituting the *KUKA Sunrise Toolbox (KST)* [37], which acts as the interface between the KUKA API and the proposed software. The robotic gripper for this study is the ReFlex TakkTile robotic hand by RightHand Robotics, a four DOF gripper with one bending DOF in each finger and one coupled rotational DOF for the two adjacent fingers [38].

C. Software Components

We designed the software using the ROS Kinetic framework (Ubuntu Xenial 16.04) as shown in Fig. 2. The ROS nodes are written in the high-level programming languages Python and MATLAB as they are most suitable for novice programmers [39]. The *Leap Hand Node* is a package that translates the fingertip positions into actuation commands for the robotic hand by measuring the angle of the fingers with respect to the palm normal vector. The Leap Motion's firmware *Leap Daemon* interfaces with the *Leap Motion*

[1]Source code and supplementary video material available at github.com/axkoenig/leap_teleop and youtube.com/watch?v=RDbpd9d7U2k.

Node [40], which publishes the raw sensor data to the ROS network at 100Hz. The *Leap KUKA Node* is composed of a script that maps the operator's hand pose to the desired robot tool pose. We use the KUKA Sunrise Toolbox (KST) [37] to interface with the robot. The *KUKA Control Node* is a package designed around the KST. This node can also be used standalone for KUKA LBR robots, facilitating the interface with different input devices, such as joysticks. The *KUKA Control Node* runs a safety controller, calculates inverse kinematics, and sends commands to the native KUKA robot controller in joint space. The *Sunrise Toolbox Server* is a part of the KST and is deployed on the KUKA robot embedded controller.

III. Robot Hand Control

The *Leap Hand Node* directly controls the four DOF of the robotic hand by evaluating the four human finger flexion angles α, β, γ and δ. The relevant variables for the calculation of α, β and γ are displayed in Fig. 3.

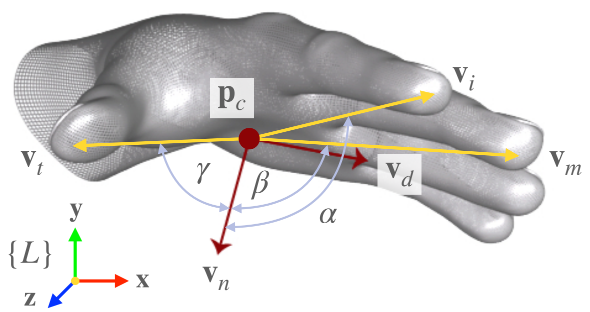

Fig. 3: Human finger flexion angles which we map to position control commands on the robotic hand [41].

Let $\mathbf{p}_c$ denote the palm center in the Leap Motion coordinate frame $\{L\}$. The finger flexion angles α, β and γ are then calculated on the basis of the following vectors from Fig. 3: the palm normal $\mathbf{v}_n$, the vector from $\mathbf{p}_c$ to the tip of the index finger $\mathbf{v}_i$, the vector from $\mathbf{p}_c$ to the tip of the middle finger $\mathbf{v}_m$ and the vector from $\mathbf{p}_c$ to the tip of the thumb $\mathbf{v}_t$. All angles are calculated with the formula in equation (1): $\alpha = f(\mathbf{v}_n, \mathbf{v}_i)$, $\beta = f(\mathbf{v}_n, \mathbf{v}_m)$, $\gamma = f(\mathbf{v}_n, \mathbf{v}_t)$.

$$f(\mathbf{v}_x, \mathbf{v}_y) = \arccos \frac{\langle \mathbf{v}_x, \mathbf{v}_y \rangle}{|\mathbf{v}_x||\mathbf{v}_y|} \tag{1}$$

To calculate the separation angle of the index and middle finger δ we apply equation (1) to two vectors pointing from $\mathbf{p}_c$ to the center of the middle segment (i.e., the middle phalanx) of the index and middle fingers. This works better in practice than using the separation angle between $\mathbf{v}_i$ and $\mathbf{v}_m$ because the human index and middle fingertips naturally approach each other when closing a grasp. This artifact would make it practically impossible to perform spherical grasps on the robotic hand. However, the middle phalanxes are more easily separable while the human hand is in a closed grasp configuration. Finally, the human flexion angles α, β, γ, and δ are linearly scaled to the allowed joint angle range of each DOF of the robot hand and sent to the ReFlex TakkTile's ROS driver.

IV. Robot Arm Control

A. Leap KUKA Node

Fig. 4 shows an overview of the main coordinates frames that are considered in the *Leap KUKA Node*. The four coordinate frames are:

- Manipulator base frame $\{B\}$
- Leap Motion frame $\{L\}$
- Hand frame $\{H\}$
- Manipulator tool frame $\{T\}$

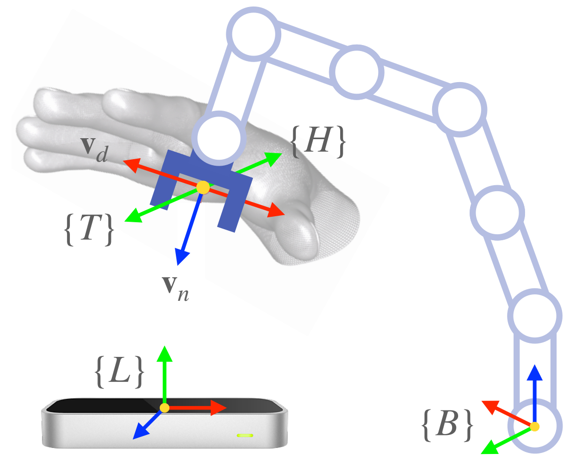

Fig. 4: Coordinate systems overview [41], [42]. We show x-axes in red, y-axes in green, and z-axes in blue.

Equation (2) states the relationships between the coordinate frames with homogeneous transformation matrices. The transformation matrix ${}^{B}_{T}T$ describes the desired tool pose $\{T\}$ expressed in the base frame $\{B\}$.

$${}^{B}_{T}T = {}^{B}_{L}T\,{}^{L}_{H}T\,{}^{H}_{T}T \tag{2}$$

According to the transform equation (2), the matrix ${}^{B}_{T}T$ is expressed as a product of the following transformations.

- ${}^{B}_{L}T$ describes the fixed pose of the Leap Motion $\{L\}$ relative to the robot base $\{B\}$. Note that we place $\{L\}$ in a convenient pose for the sake of the calculations – the Leap Motion's pose in the real world is arbitrary. We position $\{L\}$ underneath the center of a user-defined workspace and calculate its orientation by a mechanism that allows for arbitrary rotations around the Leap Motion's y-axis.
- ${}^{L}_{H}T$ expresses the pose of the operator's hand $\{H\}$ in the Leap Motion frame $\{L\}$. We define ${}^{L}_{H}T$ such that the x-axis of $\{H\}$ equals the palm direction $\mathbf{v}_d$, its z-axis equals $\mathbf{v}_n$, and its translational vector is $\mathbf{p}_c$.
- ${}^{H}_{T}T$ describes a fixed transformation from the hand frame $\{H\}$ to the robot tool pose $\{T\}$. The tool position coincides with the hand position $\mathbf{p}_c$, and we couple the tool orientation to the hand's orientation.

B. KUKA Control Node

The *KUKA Control Node* executes a Supervisory Control (SC) loop as described in algorithm 1, offering a ROS interface to receive desired tool transforms ${}^{B}_{T}T$ that are then

Algorithm 1: Supervisory control algorithm (* are functions from the KST [37]).

1 **while** `true` **do**
2   **if** `exit_app` **then**
3   break;
4   **if** `control_active` **then**
5   **if** `time_out` and $\neg$ `at_home` **then**
6   ptpMotion($\mathbf{q}_h$)*;
7   **else**
8       ${}^B_T T \leftarrow$ getToolTransform();
9       ${}^B_T \mathbf{p} \leftarrow$ scaleToolPosition(${}^B_T \mathbf{p}$, s);
10       ${}^B_T \mathbf{p} \leftarrow$ enforceWorkspaceBoundary(${}^B_T \mathbf{p}$);
11       $\mathbf{q}_d \leftarrow$ calcInvKinematics(${}^B_T T$);
12   **if** $\neg$ `mirroring` and `at_home` **then**
13   **if** all $q_{i,d} < q_t$ with $i \in \{1, \dots, 7\}$ **then**
14   realTimeMotion($\mathbf{q}_d$)*;
15   **else**
16   ptpMotion($\mathbf{q}_d$)*;
17   **else if** `mirroring` **then**
18   realTimeMotion($\mathbf{q}_d$)*
19   **else if** $\neg$ `control_active` and $\neg$ `at_home` **then**
20   ptpMotion($\mathbf{q}_h$)*;
21   publishRobotState();

used to control the robot in joint space $\mathbf{q} \in \mathbb{R}^7$. The control algorithm runs at 300Hz, and a soft watchdog guarantees a minimum execution frequency of 100Hz by monitoring the time since the last received tool transform ${}^B_T T$. The SC uses two motion types from the KST [37]: point-to-point (PTP) motions and real-time motions, which both transmit commands to the native KUKA robot controller in joint space.

The robotic arm is brought into a home position $\mathbf{q}_h$ before starting the SC loop. By default, the algorithm calculates the home position $\mathbf{q}_h$ such that the tool position coincides with the user-defined workspace center $\mathbf{p}_w$ and the tool z-axis points along the negative z-axis of the base frame $\{B\}$ (see tool home pose $\{T_h\}$ in Fig. 6). This downward-facing tool orientation is a convenient initial pose for robotic grasping, and the starting tool position at $\mathbf{p}_w$ ensures maximum clearance from our custom workspace boundary. A custom home position $\mathbf{q}_h$ can be specified, overriding this calculation process.

The SC is activated once the flag `control_active` in algorithm 1 is set to `true` via the ROS service `kuka/kuka_control`. In line 8, the desired tool transform ${}^B_T T$ is received from the ROS network. We then scale the tool position ${}^B_T \mathbf{p}$ by a multiplicative factor $s \in [0.1, 1.5]$ relative to the workspace center $\mathbf{p}_w$ in line 9. This factor is beneficial when using the control loop combined with the *Leap KUKA*

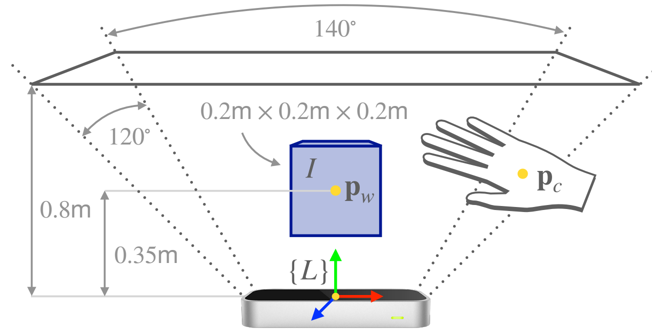

Fig. 5: Virtual interaction box I inside of Leap Motion's FOV [25], [42]. Users must place their hand inside the interaction box to start the control algorithm. We show x-axis in red, y-axis in green and z-axis in blue.

Node. A factor of $s < 1$ scales all palm movements down, allowing for higher-resolution position control of the end-effector. If $s > 1$ all movements are scaled up, allowing to cover a wider workspace on the robot with the same palm movements. In line 10, the user-defined workspace boundary constraints are enforced to ensure the tool position is within the safety bounds. The inverse kinematics problem is solved in line 11 to obtain the desired joint positions $\mathbf{q}_d$ from the updated tool transform ${}^B_T T$. Let $q_{i,d}$ be the movement of joint i from the home position $\mathbf{q}_h$ to the first desired robot position $\mathbf{q}_d$. If all joint movements $q_{i,d}$ for the first robot movement $\mathbf{q}_d$ in line 13 are below a threshold q_t (default $q_t = 0.2$ degrees) the change is small enough to directly execute a real-time motion to $\mathbf{q}_d$ in line 14. Otherwise, we perform a PTP motion to $\mathbf{q}_d$ in line 16. Now, the robot has left its home position $\mathbf{q}_h$ and the `mirroring` flag is set to `true`. From this point onwards, all calculated joint positions $\mathbf{q}_d$ are executed with real-time motions in line 18. When used with the gesture tracker, the robot now mirrors the motion of the operator's palm. Once a time-out occurs (i.e., no new transforms received for a specified period of time) or the `control_active` flag is set to `false` the robot moves to its home position $\mathbf{q}_h$ in lines 6 and 20, respectively. The algorithm continuously publishes information about the robot state in line 21 to satisfy the robot interface's real-time constraints (e.g., QoS - Quality of Service).

C. Safety Considerations

Safety is of utmost importance when letting untrained users such as children or the general public interact with robots. Fast and uncontrolled motion commands may potentially harm humans in the robot's vicinity or damage robotic equipment. We implement several features to meet the safety requirements in addition to the inherited safety of the KUKA cobot:

- We specify a maximum tool speed of 0.8 m/s, and a maximum measured cartesian torque of 25 Nm within the firmware of the KUKA iiwa. If any of those thresholds are exceeded (e.g., due to rapid hand movements or collisions), the robot performs a safety stop. The manufacturer of the robot implemented these measures.

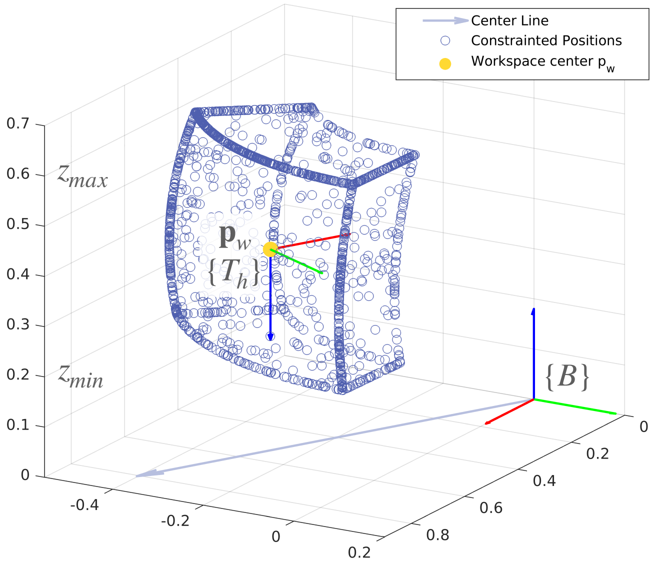

Fig. 6: Plot of random tool positions ${}^B_T\mathbf{p}$ adjusted by g. Tool home pose $\{T_h\}$ coincides with workspace center $\mathbf{p}_w$ to ensure maximum clearance from workspace boundary when algorithm starts. We show x-axes in red, y-axes in green, and z-axes in blue.

- The Leap Motion control of the robotic devices is inactive by default to prevent the system from executing any motions from hands that may be placed within the Leap Motion FOV unintentionally. A supervisor can activate the robot arm and gripper control via the ROS services `kuka/kuka_control` and `kuka/gripper_control`. Calling the emergency stop service `kuka/exit_app_emergency` immediately stops any motion of the robot.
- To start mirroring the hand's motions with the robotic arm, users must place the center of their palm $\mathbf{p}_c$ inside a virtual interaction box I. Fig. 5 illustrates that I is centered at the workspace center $\mathbf{p}_w$ which lies directly above the Leap Motion frame $\{L\}$. This mechanism ensures that the system starts hand mirroring in a well-perceived area (tracking close to Leap Motion FOV boundary is noisier). Once the robot mirrors the hand's motion, users can exit I boundaries.
- We use a moving average filter with a kernel size of 10 samples to smooth out the measured palm poses in case of swift, oscillating hand movements.
- The security software layer engages the robot breaks if any tool positions ${}^B_T\mathbf{p}$ with $z < 0$ are detected, thus preventing collision with the table.
- Operators can only change the scaling factor s via the ROS service `kuka/scaling_factor` when the Leap Motion control of the manipulator is inactive and when the robot is in its home position $\mathbf{q}_h$ with breaks engaged.
- If any problems during the inverse kinematics calculation occur (e.g., no solution or solution violates joint limits) in line 11 of algorithm 1, the robot safely stays in the previous joint configuration.
- We use the Leap Motion tracking confidence variable $c \in [0, 1]$. A low c may indicate that the hand moves very fast, is close to the FOV boundary, or that ambient lighting causes visual glare on the Leap Motion sensor. We move the manipulator to its home position $\mathbf{q}_h$ if $c < 0.05$.
- The security software layer continuously monitors the hand orientation to avoid any potentially dangerous tool orientations. We use equation (1) to compute the angle between the palm normal $\mathbf{v}_n$ expressed in $\{L\}$ and the negative Leap Motion y-axis as $\theta = f(\mathbf{v}_n, [0, -1, 0]^T)$. If $\theta > \theta_t$, where θ_t is a user-defined orientation threshold that defaults to $25°$, we clip the palm orientation to θ_t.

As a final safety layer, the tool positions ${}^B_T\mathbf{p}$ are constrained. The user-specified workspace $\mathcal{W}$ defines the set of all allowed tool positions. We implement a function $g : \mathbb{R}^3 \mapsto \mathcal{W}$ to enforce the workspace boundary, where $\mathcal{W}$ is a closed set, i.e. $\partial\mathcal{W} \in \mathcal{W}$. Equation (3) shows that g leaves all tool positions ${}^B_T\mathbf{p}$ that lie within $\mathcal{W}$ unaffected. Any tool positions that lie outside of the workspace are projected to the closest point $\hat{\mathbf{p}}$ on the workspace boundary $\partial\mathcal{W}$ as stated in equation (4).

$$g(\mathbf{p}) = \begin{cases} \mathbf{p} & \text{if} \quad \mathbf{p} \in \mathcal{W} \\ \hat{\mathbf{p}} & \text{if} \quad \mathbf{p} \notin \mathcal{W} \end{cases} \tag{3}$$

$$\text{with} \quad \hat{\mathbf{p}} \in \partial\mathcal{W} \quad \text{s.t.} \quad \min |\hat{\mathbf{p}} - \mathbf{p}| \tag{4}$$

The working envelope of the robot arm is a hollow sphere. Therefore, our user-defined workspace $\mathcal{W}$ is a segment of this hollow sphere to leverage as much of the natural robot workspace as possible and to make sure all tool positions within $\mathcal{W}$ are reachable. A simple box-shaped workspace would leave a lot of reachable tool positions unused. Fig. 6 shows a plot of a possible workspace $\mathcal{W}$. Four constraints define the workspace boundary $\partial\mathcal{W}$.

1) The *center line* defines the general orientation of the workspace (also see Fig. 6).
2) We obtain two constraining planes that are spanned by the base frame z-axis and two vectors that result from a rotation of the center line around the base frame $\{B\}$ z-axis by an an *opening angle*.
3) An *interval* $[z_{min}, z_{max}]$ spans two constraining planes in the base frame $\{B\}$ z-direction.
4) An *interval* of radii $[r_{min}, r_{max}]$ defines two spheres that are centered above the base frame $\{B\}$ and limit the workspace radially.

V. PUBLIC DEMONSTRATIONS

Fig. 7 shows one example of a possible grasping task. The operator pulls a tape that lies outside the robot's workspace closer by using a single finger and picks the tape up with a pinch grasp to place it on a basket. Such precise actions are made possible by controlling each robotic finger separately.

As a case study demonstration, we presented the robotic platform to the public during the Great Exhibition Road Festival 2019, a public science fair crowded with children and young adults, as shown in Fig. 8. We found that the robotic

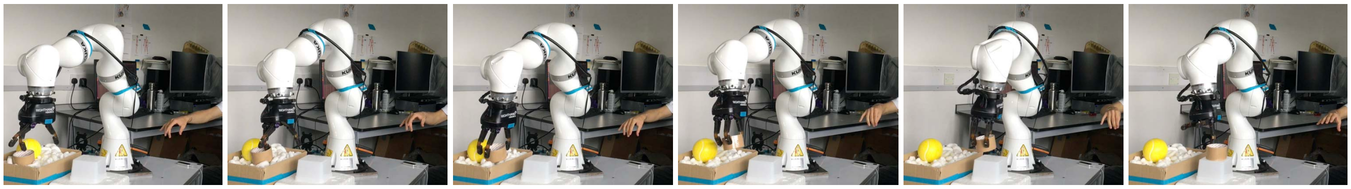

Fig. 7: Demonstration of complex manipulation task. The operator reaches for a tape with one finger, performs a pinch grasp and places the object at a target location.

platform was easy and safe to use by children while it also offered multiple layers of complexity interesting to adults. The system's safety was assessed by the college Health and Safety officer before the event. Our targeted group at the science festival was 10 to 15-year-old children. We chose this age group because we want to inspire children close to their high-school graduation before deciding which profession to aim for. Further, we used the public demonstration to ideate a curriculum of grasping tasks from which children can learn robotics and engineering concepts. We asked them to conduct a pick-and-place task and suggested performing power and pinch grasps on the provided objects. They discovered that the power grasps provided more stability on the spherical objects, whereas the pinch grasps were more suitable for delicate objects. This exercise provided an opportunity to introduce them to some basic concepts in robotic grasping (kinematics, motion-sensing via computer vision, and grasp analysis).

A questionnaire and spot interviews were collected during the event. Some data from the festival: for groups with children under 16, the main reasons to attend the exhibition were to inspire young people about science (81%) and to have an enjoyable day out (55%). This group represents 42% of all attendants to the festival. As general feedback, we achieved a high engagement from both children and adults. Here some reported comments: *"Being able to talk to researchers at a level that matters to an interested adult while also seeing the same researcher explain their work through activities that enthuse children."* and *"The whole atmosphere was great, things to do for the kids and new things to learn for the adults. I really enjoyed speaking to the presenters, who were so knowledgeable and passionate - they were truly inspiring."*.

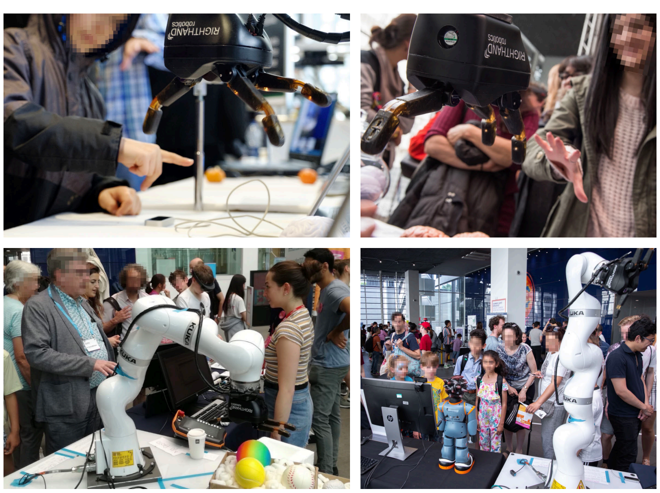

Fig. 8: Children and adults interacting with robotic platform at Great Exhibition Road Festival 2019.

VI. Conclusion

This case study introduces an open-source software framework created as an educational and demonstrative tool, specifically to engage children and young adults in different aspects of robotics. The robotic platform was designed for teleoperation with a master interface based on a hand tracking system. We have shown that an operator can simultaneously control a 7-DOF robotic end-effector with an attached 4-DOF robotic hand using palm and finger movements, creating a natural interface compared to traditional joysticks or pre-programmed trajectories. The robotic system was evaluated during a public exhibition, where it was shown to be an inspiring tool for both children and young adults. In future work, we would like to conduct extended user studies to quantify the system's effect on engagement with robotics and STEM in children and young adults. Such a quantitative analysis was not possible yet due to the outbreak of the COVID-19 pandemic. Moreover, we want to organize pilot studies that use this system to teach robotics concepts to children and young adults. We would also like to investigate the gesture control's effectiveness compared to other control techniques. Finally, this project's scope could be expanded by leveraging the system to study human grasping to define new machine learning algorithms for autonomous pick-and-place operations.

References

[1] International Federation of Robotics, "World robotics report 2020," 2020.

[2] Oxford Economics, "How robots change the world," 2019. [Online]. Available: https://www.oxfordeconomics.com/recent-releases/how-robots-change-the-world

[3] European Union, "Special eurobarometer 427 autonomous systems," 2015.

[4] M. Kont and M. Alimardani, "Engagement and mind perception within human-robot interaction: A comparison between elderly and young adults," in *Social Robotics*, A. R. Wagner, D. Feil-Seifer, K. S. Haring, S. Rossi, T. Williams, H. He, and S. Sam Ge, Eds. Cham: Springer International Publishing, 2020, pp. 344–356.

[5] M. Kandlhofer and G. Steinbauer, "Evaluating the impact of educational robotics on pupils' technical- and social-skills and science related attitudes," *Robotics and Autonomous Systems*, vol. 75, pp. 679–685, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889015001955

[6] A. Tsagaris, M. Chatzikyrkou, and I. Simeli, "Educational robotics. the pleasure of participation," *Journal of Contemporary Education, Theory & Research*, vol. 3, no. 1, pp. 31–36, 2019.

[7] S. V. Coxon, "The malleability of spatial ability under treatment of a first lego league-based robotics simulation," *Journal for the Education of the Gifted*, vol. 35, pp. 291 – 316, 2012.

[8] M. Apiola, M. Lattu, and T. A. Pasanen, "Creativity and intrinsic motivation in computer science education: Experimenting with robots," in *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education*, ser. ITiCSE '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 199–203. [Online]. Available: https://doi.org/10.1145/1822090.1822147

[9] M. M. McGill, "Learning to program with personal robots: Influences on student motivation," *ACM Trans. Comput. Educ.*, vol. 12, no. 1, Mar. 2012. [Online]. Available: https://doi.org/10.1145/2133797.2133801

[10] M. F. Yesharim and M. Ben-Ari, "Teaching computer science concepts through robotics to elementary school children," *International Journal of Computer Science Education in Schools*, vol. 2, no. 3, 2018.

[11] A. Sullivan, M. Elkin, and M. U. Bers, "Kibo robot demo: Engaging young children in programming and engineering," in *Proceedings of the 14th International Conference on Interaction Design and Children*, ser. IDC '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 418–421. [Online]. Available: https://doi.org/10.1145/2771839.2771868

[12] J.-M. Sáez-López, M.-L. Sevillano-García, and E. Vazquez-Cano, "The effect of programming on primary school students' mathematical and scientific understanding: educational use of mbot," *Educational Technology Research and Development*, vol. 67, no. 6, pp. 1405–1425, 2019.

[13] J. Vega and J. M. Cañas, "Pibot: An open low-cost robotic platform with camera for stem education," *Electronics*, vol. 7, no. 12, 2018. [Online]. Available: https://www.mdpi.com/2079-9292/7/12/430

[14] ——, "Pybokids: An innovative python-based educational framework using real and simulated arduino robots," *Electronics*, vol. 8, no. 8, 2019. [Online]. Available: https://www.mdpi.com/2079-9292/8/8/899

[15] P. Voštinár, "Using mbot robots for the motivation of studying computer science," in *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*, 2020, pp. 653–657.

[16] G. Karalekas, S. Vologiannidis, and J. Kalomiros, "Europa: A case study for teaching sensors, data acquisition and robotics via a ros-based educational robot," *Sensors*, vol. 20, no. 9, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/9/2469

[17] J. Barnes, S. M. FakhrHosseini, E. Vasey, C. H. Park, and M. Jeon, "Child-robot theater: Engaging elementary students in informal steam education using robots," *IEEE Pervasive Computing*, vol. 19, no. 1, pp. 22–31, 2020.

[18] S. Manzoor, R. Ul Islam, A. Khalid, A. Samad, and J. Iqbal, "An open-source multi-dof articulated robotic educational platform for autonomous object manipulation," *Robotics and Computer-Integrated Manufacturing*, vol. 30, no. 3, pp. 351–362, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584513001002

[19] M. Hernández-Ordoñez, M. A. Nuño-Maganda, C. A. Calles-Arriaga, O. Montaño-Rivas, and K. E. Bautista Hernández, "An education application for teaching robot arm manipulator concepts using augmented reality," *Mobile Information Systems*, vol. 2018, 2018.

[20] D. Saenz Zamarrón, N. I. Arana de las Casas, E. García Grajeda, J. F. Alatorre Ávila, and N. A. Jorge Uday, "Educational robot arm platform," *Computación y Sistemas*, vol. 24, no. 4, 2020.

[21] J. M. K. Westlund and C. Breazeal, "Transparency, teleoperation, and children's understanding of social robots," in *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2016, pp. 625–626.

[22] C. L. van Straten, J. Peter, and R. Kühne, "Child–Robot Relationship Formation: A Narrative Review of Empirical Research," *International Journal of Social Robotics*, vol. 12, no. 2, pp. 325–344, 2020. [Online]. Available: https://doi.org/10.1007/s12369-019-00569-0

[23] K. Doherty and G. Doherty, "Engagement in hci: Conception, theory and measurement," *ACM Comput. Surv.*, vol. 51, no. 5, Nov. 2018. [Online]. Available: https://doi.org/10.1145/3234149

[24] T. Vuletic, A. Duffy, L. Hay, C. McTeague, G. Campbell, and M. Grealy, "Systematic literature review of hand gestures used in human computer interaction interfaces," *International Journal of Human-Computer Studies*, vol. 129, pp. 74–94, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1071581918305676

[25] Ultraleap. Leap motion controller product page. [Online]. Available: https://www.ultraleap.com/product/leap-motion-controller/

[26] D. Bassily, C. Georgoulas, J. Guettler, T. Linner, and T. Bock, "Intuitive and adaptive robotic arm manipulation using the leap motion controller," in *ISR/Robotik 2014; 41st International Symposium on Robotics*, 2014, pp. 1–7.

[27] D. Krupke, L. Einig, E. Langbehn, J. Zhang, and F. Steinicke, "Immersive remote grasping: Realtime gripper control by a heterogenous robot control system," in *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology*, ser. VRST '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 337–338. [Online]. Available: https://doi.org/10.1145/2993369.2996345

[28] I. Jang, J. Carrasco, A. Weightman, and B. Lennox, "Intuitive bare-hand teleoperation of a robotic manipulator using virtual reality and leap motion," in *Towards Autonomous Robotic Systems*, K. Althoefer, J. Konstantinova, and K. Zhang, Eds. Cham: Springer International Publishing, 2019, pp. 283–294.

[29] S. Chen, H. Ma, C. Yang, and M. Fu, "Hand gesture based robot control system using leap motion," in *Intelligent Robotics and Applications*, H. Liu, N. Kubota, X. Zhu, R. Dillmann, and D. Zhou, Eds. Cham: Springer International Publishing, 2015, pp. 581–591.

[30] L. Peppoloni, F. Brizzi, C. A. Avizzano, and E. Ruffaldi, "Immersive ros-integrated framework for robot teleoperation," in *2015 IEEE Symposium on 3D User Interfaces (3DUI)*, 2015, pp. 177–178.

[31] Haiyang Jin, Liwei Zhang, S. Rockel, Jun Zhang, Ying Hu, and Jianwei Zhang, "A novel optical tracking based tele-control system for tabletop object manipulation tasks," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 636–642.

[32] S. Hameed, M. A. Khan, B. Kumar, Z. Arain, and M. Hasan, "Gesture controlled robotic arm using leap motion," *Indian Journal of Science and Technology*, vol. 10, no. 45, 2017.

[33] N. Epley, A. Waytz, and J. Cacioppo, "On seeing human: a three-factor theory of anthropomorphism." *Psychological review*, vol. 114 4, pp. 864–86, 2007.

[34] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*, Kobe, Japan, May 2009.

[35] J. Guna, G. Jakus, M. Pogacnik, S. Tomazic, and J. Sodnik, "An analysis of the precision and reliability of the leap motion sensor and its suitability for static and dynamic tracking," *Sensors (Basel, Switzerland)*, vol. 14, pp. 3702–20, 02 2014.

[36] KUKA. Lbr med product page. [Online]. Available: https://www.kuka.com/en-se/industries/health-care/kuka-medical-robotics/lbr-med

[37] M. Safeea and P. Neto, "Kuka sunrise toolbox: Interfacing collaborative robots with matlab," *IEEE Robotics Automation Magazine*, vol. 26, no. 1, pp. 91–96, March 2019.

[38] RightHand Robotics. Reflex product page. [Online]. Available: https://www.labs.righthandrobotics.com/reflexhand

[39] H. Fangohr, "A comparison of c, matlab, and python as teaching languages in engineering," in *Computational Science - ICCS 2004*, M. Bubak, G. D. van Albada, P. M. A. Sloot, and J. Dongarra, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 1210–1217.

[40] ROS Drivers. Code repository for leap motion ros driver. [Online]. Available: https://github.com/ros-drivers/leap_motion

[41] Ultraleap. Hand image. [Online]. Available: https://developer-archive.leapmotion.com/documentation/csharp/devguide/Leap_Hand.html

[42] ——. Leap motion image. [Online]. Available: https://blog.leapmotion.com/designing-leap-motion-controller/