

# FastOrient: Lightweight Computer Vision for Wrist Control in Assistive Robotic Grasping\*

Mireia Ruiz Maymó, Ali Shafti, and A. Aldo Faisal

**Abstract**—Wearable and Assistive robotics for human grasp support are broadly either tele-operated robotic arms or act through orthotic control of a paralyzed user’s hand. Such devices require correct orientation for successful and efficient grasping. In many human-robot assistive settings, the end-user is required to explicitly control the many degrees of freedom making effective or efficient control problematic. Here we are demonstrating the off-loading of low-level control of assistive robotics and active orthotics, through automatic end-effector orientation control for grasping. This paper describes a compact algorithm implementing fast computer vision techniques to obtain the orientation of the target object to be grasped, by segmenting the images acquired with a camera positioned on top of the end-effector of the robotic device. The rotation needed that optimises grasping is directly computed from the object’s orientation. The algorithm has been evaluated in 6 different scene backgrounds and end-effector approaches to 26 different objects. 94.8% of the objects were detected in all backgrounds. Grasping of the object was achieved in 91.1% of the cases and has been evaluated with a robot simulator confirming the performance of the algorithm.

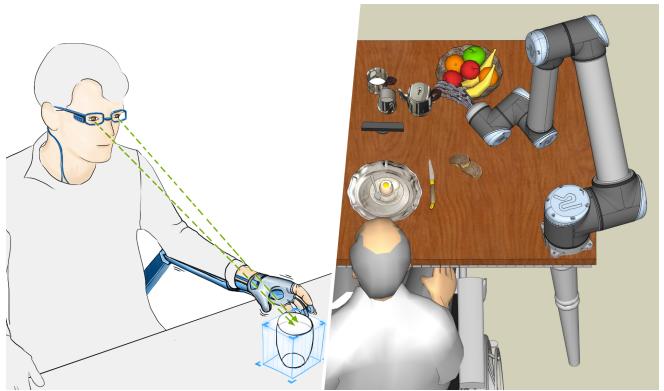
## I. INTRODUCTION

The human upper limb enables reaching, grasping and manipulations. Patients with paralysis, be it e.g. due to spinal cord injuries or neurodegenerative diseases, or amputees suffer from severe actuation limitations in daily life, leading to difficulties in reaching and/or grasping. Assistive robotic devices help in these cases, providing support to the patient’s now limited actuation capabilities. These devices, however, require low-level control by the end-user, either through residual motion (e.g. head controlled computer mouse [1]) or through neural interfaces [2], [3]. The former is not always feasible, depending on the level of paralysis or amputation, and the latter is limited by the number of independent channels that can be read out (typically outnumbered by the degrees of freedom within wearable or assistive robotic devices). We need solutions to obtain and decode human action intentions from other sources, and feed them into the control loop of the robotic device for intuitive and natural interaction.

Different modalities have been used for this, with the aim to enhance the way such devices interact with human users. Multi-modal systems relying on unaffected abilities, e.g. the use of gaze-based robotic end-point control for reaching

\*Research supported by eNHANCE (<http://www.enhance-motion.eu>) under the European Union’s Horizon 2020 research and innovation programme grant agreement No. 644000.

M. R. Maymó, A. Shafti and A. A. Faisal are with the Brain and Behaviour Lab, Dept. of Computing and Dept. of Bioengineering, Imperial College London, SW7 2AZ, London, UK. a.shafti, a.faisal@imperial.ac.uk



**Fig. 1:** Concepts of assistive robotic devices for reaching and grasping. Left: Robotic orthotic glove. Right: Tele-operated robotic hand. Both require wrist rotation control for suitable grasping.

assistance [4]–[6], or the use of wearable robotics controlled through eye winks and voice [7]. In all these cases, a suitable eventual grasp is only possible with the correct orientation of the hand; be it the human hand using an orthotic (Figure 1-left), or a tele-operated robotic hand, performing the grasp (Figure 1-right).

A third option is to use the context of the user and the robotic actuator to infer the most likely action intention, be it from a dictionary of human activity [8] or the dynamics of the preceding movements [9]. This paper describes *FastOrient*, a compact computer vision-based algorithm for the control of a robotic wrist’s rotation to obtain a suitable grasp of the object of interest. The *FastOrient* algorithm effectively off-loads complicated low-level details of grasping orientation to sub-systems that provide automated orientation adjustment, thereby reducing cognitive load and training time for the user, and thus ultimately user uptake and technology embodiment [10].

## II. BACKGROUND

Camera based approaches for end-effector orientation control have been deployed in robotics. Typically, a camera positioned on top of the end-effector or on the user’s head to detect certain characteristics of the target object. These include multi-camera approaches that segment and identify spherical and cylindrical objects in a fixed workspace[11]: they combine an eye-tracking device to detect the user’s gaze point and identify the target object and an RGBD camera positioned in a fixed mount that allows extracting depth and RGB information from the environment. Final robotic hand position and grasp depends on object shape and orientation. Similarly in [12], where end-effector camera and object segmentation are used for object boundary extraction

and detection of orientation. In [13], the shape, size and orientation of target objects are estimated using a camera located at the user's head level and the algorithm extracts features by approximating the object with geometrical models. In [14] computer vision techniques are developed to locate target objects and estimate their size and orientation using a camera and an ultrasound sensor. Wrist rotation control using image segmentation of a grayscale filtered image was added to this system [15]. In autonomous robotics [16] proposed a machine-learned algorithm that identifies points for grasping using two or more images of the same object acquired by an end-effector camera. The more general pixel-to-torque problem (using cameras to control robots) is currently only feasible in very large scale simulation settings of robot grasping and vision using deep learning [17], [18]. However, these approaches currently lack interpretability and may not translate well to safety-critical human-in-the-loop and wearable robotics settings.

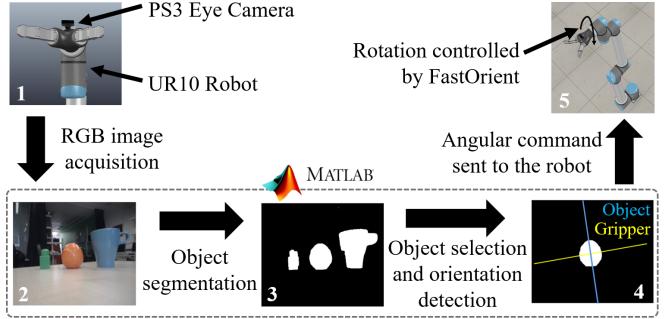
It has been shown by [19], [20] that humans tend to grasp objects along their long axis. This grasp allows taking the object with high stability and enables correct grasping in most cases. Thus, grasping orientation is not important for spherical or rounded objects, as all their axes are equal. Our assumption is that reaches often terminate in grasps and the object's that we can obtain the correct orientation of the wrist from the principal axes of the object. This forms the basis for the method described in this paper, which aims to achieve suitable grasping through a computationally light-weight and compact algorithm.

### III. METHODOLOGY

Wrist orientation control is the key step to providing a suitable grasp of an object of interest. The general architecture of our algorithm, titled *FastOrient*, is presented in Figure 2. First, an RGB image is acquired with a PlayStation 3 Eye Camera (PS3 Eye). Then, different image processing techniques are applied to the image to detect and identify the target object of interest. The detected object's orientation is then computed with reference to the corresponding gripper's orientation. This orientation is sent to the robot controller to rotate its end-effector and allow a suitable grasp of the object. For the purpose of this paper, a Universal Robots UR10 arm is used in simulation - this is interchangeable with any robotic system that allows wrist rotation.

*FastOrient* is implemented using MATLAB (R2015b 32-bit) with functions from its Image Processing Toolbox. VRep (VRep PRO EDU 3.3.1 32bit Version), a robot simulation environment, is used to reproduce and evaluate the algorithm on a robotic system. The general structure of *FastOrient* can be seen in Algorithm 1. Note that 15 frames are obtained for analysis, this is to increase robustness by considering the median orientation, avoiding effects arising from potentially deviated detections.

1) *Image acquisition*: The PS3 Eye is a digital camera producing 640x480 pixel images at a 60Hz frame rate. It integrates an infrared filter and two adjustable fixed-focus zoom lenses. The 75-degree field of view lens was selected



**Fig. 2:** General architecture of *FastOrient*. (1) Camera on top of the UR10 gripper; (2) RGB image during approaching movement; (3) Segmented image with detected objects; (4) Target object identification and orientation computed; (5) UR10 Robot in VRep simulator environment.

to enable acquisition of planes at long distances and avoid blurred objects within the image. The camera is positioned on top of the end-effector of the device. This is a more intuitive positioning than those proposed by [11], [13] as it allows to get the "hand's point of view" during reaching movement. This positioning is more affected by motion noise but ensures the object of interest is located in the camera's field of view.

2) *Color spaces*: Shadows can be identified as parts of the object itself and illumination spots may impair the complete detection of the item of interest. [21] reports that shadows share chromaticity values with the background but the brightness of pixels corresponding to these shaded areas is considerably lower. Accordingly, the Hue-Saturation-Value (HSV) space is chosen here to account for these differences. The Red-Green-Blue (RGB) model cannot separate chromaticity from brightness, while in HSV, the *Value* channel represents the brightness and, the *Hue* and *Saturation* channels together give rise to chromaticity [21]. Different thresholds can be directly applied to the *Saturation* and the *Value* channels for the removal of shadows and illumination effects. The application of thresholds in the HSV channels removes the effects of shadows but also limits the detection of grayscale objects. As a result, pre-processing of the image in the RGB space is needed. The grayscale parts of the image are thus painted in a different color, in RGB and before projection to HSV, so that they can be easily detected later. The algorithm

---

#### Algorithm 1 *FastOrient*

---

```

1: Initialise camera;
2: for  $i = 0$  to 15 do
3:   Acquire RGB image;
4:   Paint grayscale pixels green;
5:   Project image into HSV colourspace;
6:   Select target object via minimum distance algorithm;
7:   Extract features;
8:   Calculate object orientation;
9: end for
10: Calculate median orientation;
11: Calculate suitable gripper orientation;
12: return estimated angle to robot;

```

---

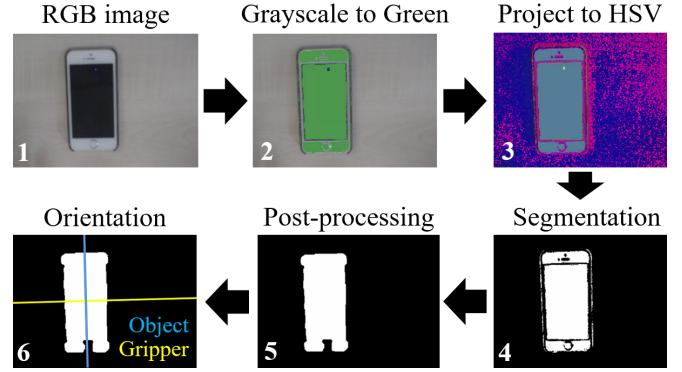
counts the number of black (RGB code: [0, 0, 0]) or dark gray (RGB code range considered: [95-125, 95-125, 95-125]) pixels of the acquired RGB image (see Figure 3.1). This is done to distinguish the cases in which black or dark backgrounds occupy a large part of the image. If this is the case, the pixel colors should remain unchanged so as to avoid the background being selected as the object. Accordingly, if more than 100,000 pixels are considered dark (the acquired images have 307,200 pixels, i.e. 32.5%), the algorithm does not change their color. However, it will look for white pixels and change their color to green, so that white objects can be detected in dark backgrounds. Nevertheless, if the number of dark pixels is below the threshold, the algorithm paints different ranges of grayscale pixels: black, gray and white, in green color. This is done by directly changing the colour of the pixel to the RGB value [77, 153, 77], which corresponds to green, as seen in Figure 3.2. Finally, a new RGB image is built with the corresponding color modifications so that the image segmentation can start with all the information concerning grayscale objects.

**3) Image segmentation:** The main goal of the image segmentation procedure is to convert the image into information easier to analyse.

**Binariation:** As seen in Figure 3.4, this consists of the transformation of the original image into a black and white one. In the binary image, the object pixels are white and the background is represented in black. This conversion is performed by applying a threshold on the HSV values to separate the object's pixels from the background ones considering each channel's histogram. Otsu's method is applied to find the optimal threshold for each image. This method assumes the image contains two classes of pixels, foreground and background, and considers the gray-level histogram of each class distribution. Then, it defines the optimal threshold that minimizes the overlap between both distributions and allows a better discrimination between the classes [22], [23]. Better results were observed if only the *Saturation* and the *Value* channels were thresholded and *Hue* information was not considered for image segmentation. Finally, the information of both channels is combined using an intersection (logical and) operation.

**Post-processing of the Binarised Image:** In order to improve the quality of object detection and make the detected shape more precise, different post-processing methods are applied (see Figure 3.5). Areas smaller than a threshold of 1500 pixels (value found empirically) are removed with an opening function. This is done to remove all the small areas of the detection that do not correspond to the object but to background or shadows. Holes in the detected parts are filled using the `imfill` MATLAB function. A combination of opening and closing functions are also applied to remove noise and make shapes smoother using the `imopen` and `imclose` MATLAB functions.

**4) Feature Extraction:** The different properties from the objects detected in the image have been measured using the `regionprops` MATLAB function. The features extracted from the objects are as follows. The centroid: `regionprops`



**Fig. 3:** Computer vision pipeline for complete target detection procedure. (1) RGB camera image, (2) Grayscale pixels detected and painted in green, (3) Projection into HSV color space, (4) Image segmentation for analysis of object's features, (5) Post-processing to improve detection quality, (6) Object's orientation (blue) and gripper's orientation (yellow) calculated.

calculates the center of mass of the region used to calculate the minimum distance algorithm. The bounding box: `regionprops` gives the smaller rectangle containing the region used to create a mask for target object discrimination. The orientation: `regionprops` returns a scalar corresponding to the angle of the long axis of an ellipse that has the same second moment of inertia as the object region, with respect to the horizon.

**5) Target object detection:** To discriminate the target object from all the other objects that are detected in the image, the same procedure as [14] is used, selecting the object located closest to the center of the image.

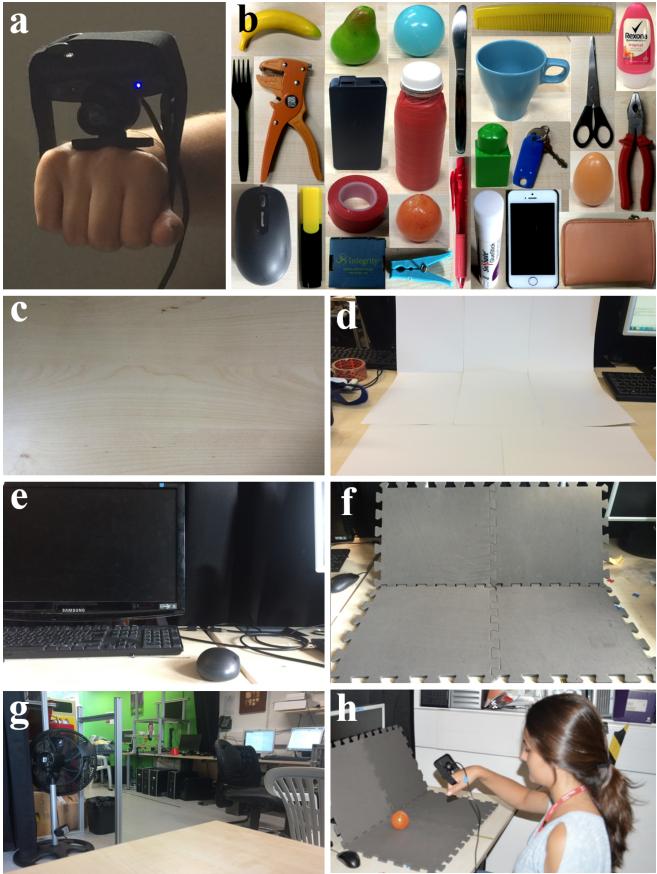
**6) Robotic end-effector orientation calculation:** The orientation of the detected object is calculated for each obtained frame and the median is computed. Then, the perpendicular orientation is calculated for the end-effector orientation in order to place the gripper along the long axis of the object (see Figure 3.6). This orientation is sent as a command to the robotic arm as the output of *FastOrient*.

#### IV. EVALUATION EXPERIMENTS AND RESULTS

The evaluation is divided into three sections: (1) detection of the target object, (2) the object and gripper orientation as returned by *FastOrient*, and (3) the rotation of the robotic gripper in VRep, the robotic simulation environment. The evaluation is performed in six different experiment setups considering 26 different objects with each approach repeated ten times.

##### A. General conditions

To simulate the reaching movement, the PS3 Eye is positioned on top of an operator's right closed fist to collect image data from natural human reaches. The base of the camera is in contact with the back of their hand and the lens is focusing forwards (see Figure 4.a). A total of 26 different objects are tested for the evaluation of the control algorithm (see Figure 4.b). These are objects used in activities of daily living, with different shapes, colors and materials. Five different backgrounds are defined to evaluate the system. These backgrounds represent almost all possible daily situations for



**Fig. 4:** Evaluation experiment setup and definitions. (a) Camera placement on operator’s hand to collect image data from natural human reaches, (b) our 26 objects used for evaluation, various backgrounds and surfaces used for evaluating robustness: (c) Beige table background, (d) White background, (e) Dark background, (f) Desk background, (g) Open lab background, (h) Evaluation setup with the operator reaching for the object of interest.

object grasping (see Figure 4.c-g). The lights in the lab are on during the experiments and the curtains are open in order to have natural light.

Reaching approaches are divided into four types, (1) Top approach: is defined as being perpendicular to the surface where the object is positioned; (2) Lateral approach: is parallel to this surface; (3) Angulated approach: is at approximately 60 degrees with respect to the surface, and (4) Variable approach: represents the most natural approach for each object, as preferred by the human operator. The arm trajectory follows a straight line from its initial position (approximately 35cm from the object) to its final point (approximately 20cm from the object). Combinations of the reaching approach types and backgrounds are then selected as follows to define six experiment setups: 1. Beige Table + Top, 2. White + Top/Lateral (depending on the object), 3. Desk + Lateral, 4. Dark + Top/Lateral (depending on the object), 5. Open Lab + Angulated (Open Lab 1), and 6. Open Lab + Variable (Open Lab 2).

#### B. Evaluation of target object selection

Target object selection is evaluated for all objects and backgrounds. In order to assess performance, the output

image obtained after the segmentation process is classified through visual inspection, into categories as follows. *Complete detection*: output shape corresponds exactly to the real shape of the object; *Partial detection affected by shadows*: output shape is not the same as the real one, some parts are added due to effects of shadows; *Partial detection affected by illumination*: output shape is incomplete, some parts are missing due to effects of illumination; *Partial detection affected by gray parts*: output shape is incomplete, all the parts are detected except for the grayscale parts in the object; *None*: the object is not detected at all.

#### C. Evaluation of object orientation detection

The output of the object orientation detection, shown visually as a line drawn along the object’s axis as detected on the image, is also evaluated through visual inspection. The returned orientation is then classified as follows. *Complete alignment* with the object’s long axis; *Affected by partial detection* where the detected orientation deviates from the long axis by  $< 20^\circ$ , grasping is still allowed; *Incorrect* where deviation is  $> 20^\circ$ , grasping is not possible; *None*, where no orientation is retrieved due to the object not being detected in the previous step.

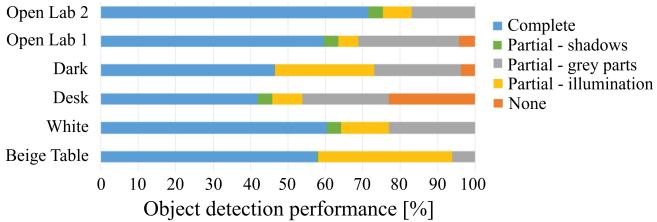
#### D. Evaluation with robot simulation

VRep is an environment for robot simulations. The robot used for the simulation is the Universal Robots UR10 robot, an anthropomorphic 6 degrees-of-freedom robotic arm, equipped with a two finger gripper as its end-effector. Within the simulation environment, a vision sensor is added between the gripper fingers to simulate the camera view. The VRep simulation is controlled via MATLAB through VRep’s internal APIs. The aim of the interface is to communicate the *FastOrient* orientation angle from MATLAB to VRep.

Five different environments are simulated, replicating the different backgrounds tested. Five different objects with different shapes and colors are also selected for this (banana, orange, red bottle, blue mug and red tool in Figure 4.b). Each object is positioned in one of the experiment backgrounds and for each of them, three different reaching approaches have been tested: Top, Lateral and Angulated. The final orientation of the gripper with respect to the object and the view from the vision sensor will confirm whether grasping is possible in the simulation and consequently, if the algorithm has worked correctly.

#### E. Results and discussion

1) *Target object selection performance*: Considering a general detection category which also includes partial detections, 94.8% of cases are detected by *FastOrient*. In particular, all items are detected with the Beige Table, White and Open Lab 1 background. 96% of the objects are detected for the Dark and Open Lab 1 cases. A smaller percentage is detected for the Desk background (77%) due to the presence of a large number of black pixels, which prevents the algorithm from painting grayscale objects in green and consequently affects their detection. Across all



**Fig. 5:** Detailed object detection performance results in percentages of complete detection, partial detection due to shadows, illumination or grayscale parts in objects and non-detected items, for the six different experiment setups.

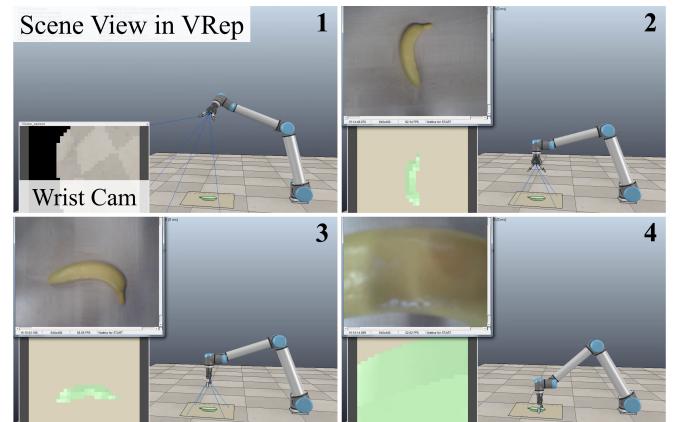
experiments and repetitions, 56.3% of cases are completely detected. However, 2.63%, 16% and 19.9% of the detections are affected by shadows, illumination or the presence of grayscale parts in objects, respectively. 5.19% of the objects are not detected at all. Full results can be seen in Figure 5).

2) *Object orientation detection performance:* Results show that all the complete detection cases result in complete orientations clearly aligned with the object's long axis. For the cases of partial detection affected by shadows, the majority of the cases give an aligned orientation where grasping is still allowed (70%) while the others are defined as incorrectly oriented. Although the effect of illumination and the presence of grayscale parts may give a partial detection, the implication in the orientation retrieved from *FastOrient* depends on the proportion of the area detected. Approximately 50% of the cases affected by illumination still retrieve a correct orientation, 35% result in slightly deviated orientations and 15% have a wrong orientation. For the partial detection caused by grayscale parts, approximately 55% of the cases result in a completely aligned orientation, 40% are affected and give an orientation within the twenty degrees margin of error and 5% of the cases result in a wrong orientation. Obviously, all the cases where the object is not detected lead to no orientation returned.

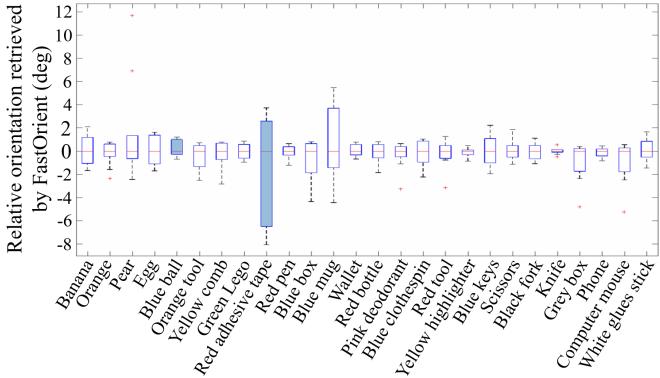
3) *Robot simulation performance:* To test if the orientation of the gripper is the appropriate one to achieve the grasping of an object, simulations using the VRep environment and the UR10 robot are performed. In these simulations, the gripper is rotated by *FastOrient*. The final configuration of the gripper needs to have its fingers aligned with the yellow line (e.g. in Figure 3.6). The robot's other joints are actuated so that the gripper gets close to the object to evaluate if the grasping is correct. The vision sensor added on the gripper allows to have visual feedback from the simulated robot movement. The example of a banana approached from the top is shown in Figure 6, to visualize how the simulation works. During the approaching movement, the corresponding end-effector orientation is calculated by *FastOrient*. This angle is sent to the VRep environment and the gripper is rotated (as shown in Figure 6.3). Then, a faster approach takes place to position the gripper close to the object, Figure 6.4. From this final configuration of the robot, the positioning of the gripper gives information of the grasping. If its fingers are perpendicular to the long axis of the object, as for the banana in Figure 6.4, the object can be easily picked up.

4) *Repeatability:* Boxplots are drawn for each object and experiment to represent the variability of each orientation detection and consequently, study the reproducibility of the system. Although the object is not always positioned at orientation zero (horizontal line) in the setups, relative angles are computed so that all the variations are started from angle zero. The different computed orientations are set to be between  $-90^\circ$  and  $90^\circ$ . The performance of the system is precise and robust for all the experiments. It can be seen in the corresponding boxplots that, for most of the cases, the variability for the orientation given for the 10 repetitions is within a  $4^\circ$  range or less for all experiments. This small variability can be due to small effects of illumination and shadows or to other effects such as the instability inherent to the approaching movement. For each experiment, the round objects' boxplots are the ones that show a higher variability between repetitions and have been marked in blue to differentiate them from the rest. This is because round objects do not have a clear orientation and hence, the algorithm's variability is justified. The boxplot for the "Open Lab 2" experiment setup is included as an example in Figure 7.

Across our experiments, the variability for each experiment is very low, confirming the good reproducibility of the system. Nonetheless, we observe outliers. Experiments 3, 5 and 6 are the more accurate ones as their outliers are comprised in the  $-2^\circ$  to  $2^\circ$  range. For the rest of experiments, the outliers are spread further within a  $-4^\circ$  to  $8^\circ$  range. Note, that most objects can be grasped reliably with a larger misalignment than observed here. Accordingly, external effects such as the time of the day and lighting conditions in which the experiments are performed do not affect the performance of the algorithm in a considerable manner as experiments are performed independently at different times. Although during the early morning and late afternoon the effect of illumination can create more shadows in our data, yet our algorithm removes them properly as no extra variability is



**Fig. 6:** Grasp orientation optimization algorithm in simulation. (1) The robot is reaching for the object location, (2) The vision sensor is used to detect the banana, its orientation and to then calculate the suitable grasp orientation, (3) The robot's wrist is rotated into the suitable grasp orientation, (4) The robot reaches in to fulfill the grasp.



**Fig. 7:** Boxplot for each object tested in the Open Lab 2 background to evaluate the hand orientation algorithm for the ten repetitions. The boxplots painted in blue correspond to round objects.

added to specific experiments.

## V. CONCLUSIONS

We presented computer vision guided control of wrist orientation for wearable and assistive robotics to help individuals with motor disabilities. Typically these devices rely on the human user's direct control for the minutiae of interaction. In our case, we demonstrated that our *FastOrient* algorithm works out-of-the-box which does not require a complex training learning pipeline and is sufficient to allow a natural wrist and grasp interaction. With the technology that we have demonstrated, we should be able to use gaze-based 3D positioning of the arm for grasping, as demonstrated in [4]–[6], wink-based detection of grasping intention [7], and now automatic processing of the grasp orientation based on the findings of this paper, without the requirement for the user to fine-tune the orientation of the hand, resulting in a full, intuitive assistive system. We have demonstrated *FastOrient* across 5 different realistic surfaces and for 26 different objects pertaining to standard activities of daily living. *FastOrient* executed successful grasps in 91.1% of the evaluated cases. While this method may not be suitable for objects with very complex shapes, it is nonetheless sufficient for the many typical objects used here. Our compact computer vision algorithm *FastOrient* can be efficiently ported to small, light, low-power embedded systems, by cross-compiling from its MATLAB sources to numerous embedded platforms. This facilitates its deployment in a variety of systems, such as upper arm prosthetics, orthotics or exoskeletons.

## REFERENCES

- [1] N. Sim, C. Gavriel, W. W. Abbott, and A. A. Faisal, “The head mousehead gaze estimation” in-the-wild” with low-cost inertial sensors for bmi use,” in *Neural Engineering (NER), 2013 6th International IEEE/EMBS Conference on*, pp. 735–738, IEEE, 2013.
- [2] S. Fara, C. S. Vikram, C. Gavriel, and A. A. Faisal, “Robust, ultra low-cost mmg system with brain-machine-interface applications,” in *Neural Engineering (NER), 2013 6th International IEEE/EMBS Conference on*, pp. 723–726, IEEE, 2013.
- [3] D. Farina and S. Amsüss, “Reflections on the present and future of upper limb prostheses,” *Expert review of medical devices*, vol. 13, no. 4, pp. 321–324, 2016.
- [4] P. M. Tostado, W. W. Abbott, and A. A. Faisal, “3D gaze cursor: Continuous calibration and end-point grasp control of robotic actuators,” *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2016-June, pp. 3295–3300, 2016.
- [5] S. Dziemian, W. W. Abbott, and A. Aldo Faisal, “Gaze-based telepresence enables intuitive continuous control of complex robot arm use: Writing & drawing,” *Proceedings of the IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics*, vol. 2016-July, pp. 1277–1282, 2016.
- [6] R. O. Maimon-Dror, J. Fernandez-Quesada, G. A. Zito, C. Konnaris, S. Dziemian, and A. A. Faisal, “Towards free 3D end-point control for robotic-assisted human reaching using binocular eye tracking,” in *Rehabilitation Robotics (ICORR), 2017 International Conference on*, pp. 1049–1054, IEEE, 2017.
- [7] B. Noronha, S. Dziemian, G. A. Zito, C. Konnaris, and A. A. Faisal, “Wink to graspcomparing eye, voice & EMG gesture control of grasp with soft-robotic gloves,” in *Rehabilitation Robotics (ICORR), 2017 International Conference on*, pp. 1043–1048, IEEE, 2017.
- [8] A. A. Thomik, D. Haber, and A. A. Faisal, “Real-time movement prediction for improved control of neuroprosthetic devices,” in *Neural Engineering (NER), 2013 6th International IEEE/EMBS Conference on*, pp. 625–628, IEEE, 2013.
- [9] M. Xiloyannis, C. Gavriel, A. A. Thomik, and A. A. Faisal, “Gaussian process autoregression for simultaneous proportional multi-modal prosthetic control with natural hand kinematics,” *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 10, pp. 1785–1801, 2017.
- [10] T. R. Makin, F. de Vignemont, and A. A. Faisal, “Neurocognitive barriers to the embodiment of technology,” *Nat. Biomed. Eng*, vol. 1, p. 0014, 2017.
- [11] D. P. McMullen, G. Hotson, K. D. Katyal, B. A. Wester, M. S. Fifre, T. G. McGee, A. Harris, M. S. Johannes, R. J. Vogelstein, and A. D. Ravitz, “Demonstration of a semi-autonomous hybrid brain-machine interface using human intracranial EEG, eye tracking, and computer vision to control a robotic upper limb prosthetic,” *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 4, pp. 784–796, 2014.
- [12] P. J. Sanz, A. P. Del Pobil, J. M. Inesta, and G. Recatala, “Vision-guided grasping of unknown objects for service robots,” in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 4, pp. 3018–3025, IEEE, 1998.
- [13] M. Markovic, S. Dosen, D. Popovic, B. Graumann, and D. Farina, “Sensor fusion and computer vision for context-aware control of a multi degree-of-freedom prosthesis,” *Journal of neural engineering*, vol. 12, no. 6, p. 66022, 2015.
- [14] S. Došen, C. Cipriani, M. Kostić, M. Controzzi, M. C. Carrozza, and D. B. Popović, “Cognitive vision system for control of dexterous prosthetic hands: experimental evaluation,” *Journal of neuroengineering and rehabilitation*, vol. 7, no. 1, p. 42, 2010.
- [15] S. Došen and D. B. Popović, “Transradial prosthesis: artificial vision for control of prehension,” *Artificial organs*, vol. 35, no. 1, pp. 37–48, 2011.
- [16] A. Saxena, J. Driemeyer, and A. Y. Ng, “Robotic grasping of novel objects using vision,” *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 157–173, 2008.
- [17] J. Redmon and A. Angelova, “Real-time grasp detection using convolutional neural networks,” in *International Conference on Robotics and Automation (ICRA)*, 2015.
- [18] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. P. Sampedro, K. Konolige, S. Levine, and V. Vanhoucke, “Using simulation and domain adaptation to improve efficiency of deep robotic grasping,” 2018.
- [19] M. R. Cutkosky, “On grasp choice, grasp models, and the design of hands for manufacturing tasks,” *IEEE Transactions on robotics and automation*, vol. 5, no. 3, pp. 269–279, 1989.
- [20] T. Feix, J. Romero, H.-B. Schmidtmayer, A. M. Dollar, and D. Kragic, “The grasp taxonomy of human grasp types,” *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 1, pp. 66–77, 2016.
- [21] M. Zhao, J. Bu, and C. Chen, “Robust background subtraction in HSV color space,” in *Multimedia systems and Applications V*, vol. 4861, pp. 325–333, International Society for Optics and Photonics, 2002.
- [22] L. Jianzhuang, L. Wenqing, and T. Yupeng, “Automatic thresholding of gray-level pictures using two-dimension Otsu method,” in *Circuits and Systems, 1991. Conference Proceedings, China., 1991 International Conference on*, pp. 325–327, IEEE, 1991.
- [23] N. Otsu, “A threshold selection method from gray-level histograms.” *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.