

2D Linear Time-Variant Controller for Human’s Intention Detection for Reach-to-Grasp Trajectories in Novel Scenes

Claudio Zito^{1,2,*}, Tomasz Deregowski^{1,*} and Rustam Stolkin²

Abstract—Designing robotic assistance devices for manipulation tasks is challenging. This work is concerned with improving accuracy and usability of semi-autonomous robots, such as human operated manipulators or exoskeletons. The key insight is to develop a system that takes into account context- and user-awareness to take better decisions in how to assist the user. The context-awareness is implemented by enabling the system to automatically generate a set of candidate grasps and reach-to-grasp trajectories in novel, cluttered scenes. The user-awareness is implemented as a linear time-variant feedback controller to facilitate the motion towards the most promising grasp. Our approach is demonstrated in a simple 2D example in which participants are asked to grasp a specific object in a clutter scene. Our approach also reduce the number of controllable dimensions for the user by providing only control on x - and y -axis, while orientation of the end-effector and the pose of its fingers are inferred by the system. The experimental results show the benefits of our approach in terms of accuracy and execution time with respect to a pure manual control.

I. INTRODUCTION

Semi-autonomous robots operated by human subjects to augment, substitute or assist motor functions are rapidly gaining ground becoming a mainstream type of technology. Their application is extensive in several domains: robots operate semi-autonomously in hazardous environments while the user is in safety, e.g. nuclear waste disposal [4], but also exoskeletons and prostheses are used by patients with neuromuscular disorders to regain motor functions [3], [5]. Though, current human-machine interfaces (HMIs) are still not capable enough to control these devices effectively and to their full potential. So it results in low take-up, high cognitive burden for the users, poor coordination, long training session and slow operation speeds. This is particularly noticeable in upper-limb robots, due to the inherent complexity and ample possibilities of arm and hand movements [3].

Our work is a preliminary investigation towards context- and user-awareness in semi-autonomous manipulators. We are concerned with how to reduce the user’s burden and the training session time for a semi-autonomous robot manipulator by making the system more intuitive for the user. In contrast to previous work, e.g. [8], this is done by adding an active AI component in the semi-autonomous system instead of solely improving the user’s feedback, e.g. [9], [14], [15], [17].

This work was supported by UK Engineering and Physical Sciences Research Council (EPSRC No. EP/R02572X/1) for the National Centre for Nuclear Robotics (NCNR).

* First co-author

¹ IRLab, School of Computer Science, University of Birmingham, UK

² ERL, School of Metallurgy and Materials, University of Birmingham, UK, C.Zito@bham.ac.uk

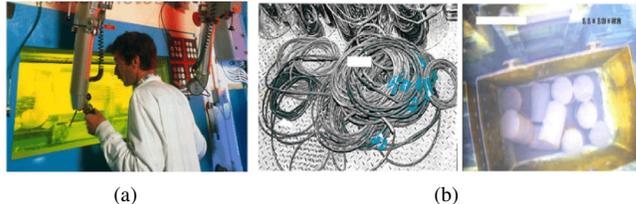


Fig. 1: The figure shows: (a) current workspace of the human operator for the sort and segregate application. The operator has to look through a small window to operate the mechanical arms; and (b) typical waste items for a nuclear facility: wiry, free-form and deformable objects. Pictures courtesy of UK National Nuclear Lab Workington test facility. Best seen in colours.

The key insight of this paper is that by making the system user- and context-aware we will provide better usability and accuracy. This work is mainly focused on grasping in cluttered scenes. We improve context-awareness by employing a generative model similar to [7], [12], [16] to estimate a set of possible candidate grasps in the scene, and for each candidate grasp we plan a reach-to-grasp trajectory. The user’s motion commands are then used to detect his/hers intention and facilitate the motion towards the selected target. To do so, we proposed a linear time-variant LQR controller that filters the motion command along the candidate trajectory. In other words, the LTV-LQR creates a basin of attraction towards the most promising trajectory. In addition, our system provides the user with controllability over 2D (x - and y -axis) while the orientation of the end-effector and the pose of the fingers is automatically inferred from the selected trajectory. We also encode in the LTV-LQR a recovering method to identify when the user to move away from an undesired trajectory and guide him/her towards a new best candidate.

Although several other researchers have investigated LQR approaches for stabilising human-like motions, e.g. [1], [13], to our knowledge this is the first time that such an approach is used to identify human’s intention and to improve reach-to-grasp actions.

II. BACKGROUND

In this section we present the basic components that we use for generating grasps and approaching trajectories on a clutter scene. First, we introduce the concept of 2D surface features, which are used to learn a contact. Contacts are then encoded as kernels to build a full joint probability density

(PDF) of robot's links and of surface features. While this idea was firstly introduced by us in [7] to learn 3D contacts, in this work we present a simplified version in 2D. The contacts are a generative models that allows us to generate candidate grasps on a clutter scene. Finally we introduce the concept of an linear time-variant LQR controller that is used to create manifolds along the trajectories. Such manifolds enable us to detect the user's intention and guide him/her towards the selected target.

A. Surface features

To learn a probabilistic representation of a contact we rely on local object surface features. Features are composed of a 2D position, a 1D orientation, and a 1D local surface descriptor that encodes the local curvature. A feature belongs to the space $SE(2) \times \mathbb{R}$, where $SE(2)$ is the special Euclidian space that describes a rigid body pose by x -, y -axis and a rotation over the z -axis, θ . The surface descriptors are composed of a scalar. We thus represent an object as a set S of N_S surface features s_j

$$S = \{s_j : s_j \in SE(2) \times \mathbb{R}\}_{j \in [1, N_S]}. \quad (1)$$

Let us denote the separation of a feature s into $\mathbf{p} \in \mathbb{R}^2$ for position, a θ is an angle in radian for orientation, and $r \in \mathbb{R}$ for the surface descriptor. For compactness, we also denote the pose of a feature as v . As a result, we have $s = (v, r)$, and $v = (\mathbf{p}, \theta) \in SE(2)$.

Surface descriptors correspond to the local principal curvature [11]. The surface normal at \mathbf{p} is computed from the nearest neighbours of \mathbf{p} using a PCA-based method, e.g. [6]. The curvature at point \mathbf{p} is encoded along the plane tangential to the object's surface, r , i.e. perpendicular to the surface normal at \mathbf{p} . The surface normal and principal direction allow us to define a 2D reference frame that is associated to a point \mathbf{p} .

B. Kernel density estimation

In this paper, we represent PDFs non-parametrically with a set of features, or particles. The underlying PDF is created through *kernel density estimation* [10], by assigning a kernel function to each particle supporting the density. For contact models, we consider PDFs defined on surface features s (Sec. II-A). For that purpose, let us denote by $\boldsymbol{\mu}$ a surface feature vector given by $\boldsymbol{\mu} = (\mu_p, \mu_\theta, \mu_r)$, and by $\boldsymbol{\sigma}$ a vector of real numbers $\boldsymbol{\sigma} = (\sigma_p, \sigma_\theta, \sigma_r)$. Where $\mu_p = (\mu_x, \mu_y)$ is the position mean, and $\sigma_p = (\sigma_x, \sigma_y)$ is the diagonal of the covariance. We thus define our kernel as

$$\mathcal{K}(s | \boldsymbol{\mu}, \boldsymbol{\sigma}) = \mathcal{N}_2(\mathbf{p} | \boldsymbol{\mu}_p, \boldsymbol{\sigma}_p) \mathcal{N}_1(\theta | \mu_\theta, \sigma_\theta) \mathcal{N}_1(r | \mu_r, \sigma_r), \quad (2)$$

where μ is the kernel mean point, σ is the kernel bandwidth, \mathcal{N}_n is an n -variate isotropic Gaussian kernel. Given a set of K surface features, the probability density in a region of space is then determined by the local density of the particles in that region, as

$$P(s) \simeq \sum_{j=i}^K w_j \mathcal{K}(s | s_j, \boldsymbol{\sigma}), \quad (3)$$

where $\boldsymbol{\sigma} \in \mathbb{R}^4$ is the kernel bandwidth and $w_j \in \mathbb{R}^+$ is a weight associated to s_j such that $\sum_j w_j = 1$.

C. Linear Time-Variant LQR

Linear time-variant LQR (LTV-LQR) is a mathematically elegant solution for controlling non-linear dynamical system [2]. The underlying idea is to locally linearise the dynamics of the system around a feasible trajectory. By feasible we mean that the trajectory is a solution of the dynamics. Let us consider the non-linear dynamic system

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \quad (4)$$

and its Taylor expansion around a point $(\mathbf{x}_0, \mathbf{u}_0)$, which results in

$$\begin{aligned} \dot{\mathbf{x}} &\approx f(\mathbf{x}_0, \mathbf{u}_0) + \frac{\partial f}{\partial \mathbf{x}}(\mathbf{x} - \mathbf{x}_0) + \frac{\partial f}{\partial \mathbf{u}}(\mathbf{u} - \mathbf{u}_0) \\ &= \mathbf{c} + \mathbf{A}(\mathbf{x} - \mathbf{x}_0) + \mathbf{B}(\mathbf{u} - \mathbf{u}_0) \end{aligned} \quad (5)$$

Following the standard notation, $\mathbf{x} \in \mathbb{R}^n$ represents a state, $\mathbf{u} \in \mathbb{R}^m$ an input control, $\dot{\mathbf{x}} \in \mathbb{R}^n$ is the state change, $\mathbf{A} \in \mathbb{R}^{n \times n}$ and $\mathbf{B} \in \mathbb{R}^{n \times m}$ describe the transition dynamics of the system and how the effects of the input control over states, respectively.

By parametrising the system over time we change coordinates to

$$\begin{aligned} \bar{\mathbf{x}}(t) &= \mathbf{x}(t) - \mathbf{x}_0(t) \\ \bar{\mathbf{u}}(t) &= \mathbf{u}(t) - \mathbf{u}_0(t) \end{aligned} \quad (6)$$

in which $[\mathbf{x}_0(t), \mathbf{u}_0(t)]$ is a feasible trajectory. We can then rewrite the system as

$$\dot{\bar{\mathbf{x}}}(t) = \dot{\mathbf{x}}(t) - \dot{\mathbf{x}}_0(t) = \dot{\mathbf{x}}(t) - f(\mathbf{x}_0(t), \mathbf{u}_0(t)) \quad (7)$$

and by using Eq. 5

$$\begin{aligned} \dot{\bar{\mathbf{x}}}(t) &= \frac{\partial f_0}{\partial \mathbf{x}}(\mathbf{x}(t) - \mathbf{x}_0(t)) + \frac{\partial f_0}{\partial \mathbf{u}}(\mathbf{u}(t) - \mathbf{u}_0(t)) \\ &= A(t)\bar{\mathbf{x}}(t) + B(t)\bar{\mathbf{u}}(t) \end{aligned} \quad (8)$$

such that the linearisation of the dynamics, i.e. the reference of the coordinate system, moves along the trajectory. In Eq. 8 we used a compacted notation f_0 , instead of $f(\mathbf{x}_0(t), \mathbf{u}_0(t))$, to write the partial derivatives of f at point $(\mathbf{x}_0(t), \mathbf{u}_0(t))$.

The trajectory stabilisation is formulated as minimising the following finite horizon cost function in quadratic form,

$$\begin{aligned} J(\bar{\mathbf{x}}, 0) &= \frac{1}{2} \bar{\mathbf{x}}(0)^\top Q(0) \bar{\mathbf{x}}(0) \\ &+ \frac{1}{2} \sum_{t=N}^1 (\bar{\mathbf{x}}(t)^\top Q(t) \bar{\mathbf{x}}(t) + \bar{\mathbf{u}}(t)^\top R(t) \bar{\mathbf{u}}(t)) \end{aligned} \quad (9)$$

where the top line encodes a penalising factor for not reaching the goal state, $\mathbf{x}(0)$, at the end of the trajectory. The bottom line penalises the system at time t for being away from $\mathbf{x}_0(t)$. The state cost-weighting matrices $Q(t)$, for $t \geq 0$, need to be symmetric positive semi-definite, while the control cost-weighting matrix $R(t)$ need to be positive definite. Our implementation of the state and control cost-weighting matrices is shown in Sec. III-C.

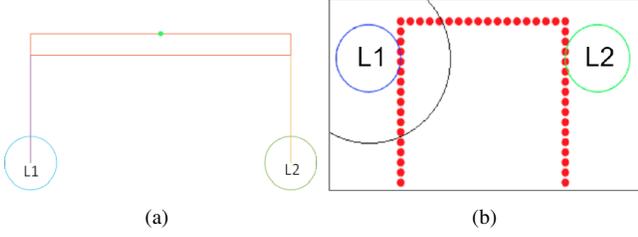


Fig. 2: (a) A simulated two spherical fingers parallel gripper. The gripper is composed by one link for each finger, L1 and L2. (b) The learnt contact model as a pinch grasp on a rectangular shaped object. The black circle around link L1 represents the neighbour in which surface's features are considered for learning the contact. Best seen in colours.

The feedback control matrix, $K(t) \in \mathbb{R}^{m \times n}$, is used to compute the optimal control input at time t that minimises the cost function in Eq. 9, such that

$$\mathbf{u}^*(t) = \mathbf{u}_0(t) - K(t)\mathbf{x}(t) \quad (10)$$

and K is computed as

$$K_t = R(t)^{-1}B(t)^\top P(t) \quad (11)$$

$P(t)$ is found by solving the Riccati Differential Equation [2].

III. OUR APPROACH

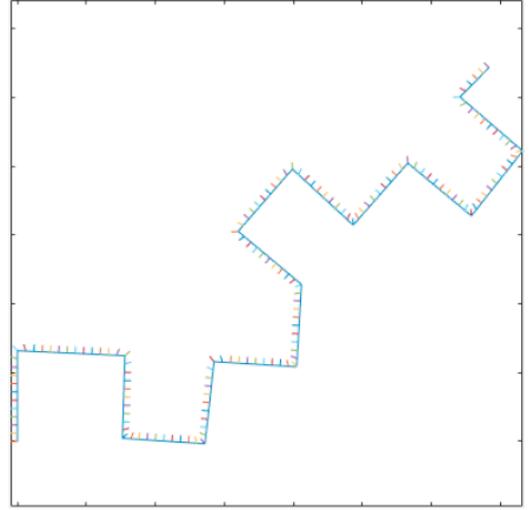
In this section we describe our approach. Given a novel scene, we generate a set of possible candidate grasps and, for each grasp, a reach-to-grasp trajectory for the gripper. We then compute a sequential feedback controller for each trajectory. When the user's input comes in, the system selects the best feedback controller that matches the input and assist the user along the selected trajectory. If the user moves away from the selected trajectory, the effects of the feedback controller gradually disappear until a new feedback controller will be selected.

A. Contact model

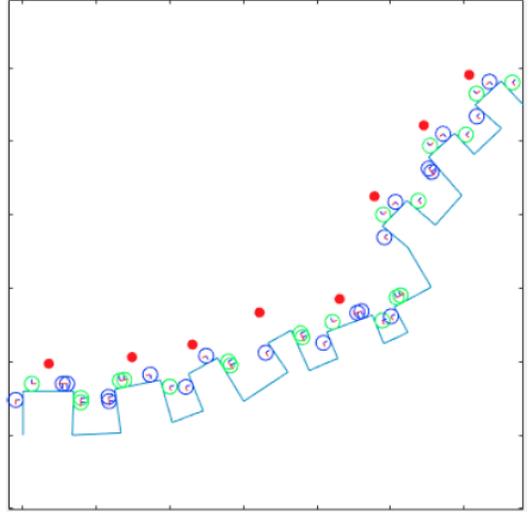
To generate a set of possible candidate grasp we first need to learn a contact model. A contact model is a full joint probability of the pose of the gripper's links and of an object's surface features. It describes the relations between the gripper's links and the object's surface involved in the contact. Figure 2 shows a graphical representation of (a) the gripper and (b) a pinch grasp contact model.

Given a 2D description of the clutter scene O as a set of N_O surface features $\{s_j\}_{j=1}^{N_O}$ (Sec. II-A), a contact model C^{L_i} is constructed for the i^{th} link. Surface features close to the link surface are considered more important than those lying far from the surface, thus the features are weighted, to make their influence on C^{L_i} decrease exponentially with the square distance to the link. Let us denote by $z_j = (\mathbf{p}_j, \theta_j)$ the pose of L relative to the pose v_j of the j^{th} surface feature. In other words, z_j is defined as

$$z_j = v_j^{-1} \circ l, \quad (12)$$



(a)



(b)

Fig. 3: Our simulated 2D scene. Figure (a) shows the landscape of a possible scene with surface's normals computed. Figure (b) shows a set of link's poses, blue circles for L1 and green for L2, generated by the query density. The red points identify the candidate grasps, as a 2D pose for the reference frame of the gripper, sampled from the query density which are configurations of the gripper where both links generate a contact. Best seen in colours.

where l denotes the pose of L_i in the world frame, \circ denotes the pose composition operator, and v_j^{-1} is the inverse of v_j . The contact model is then estimated as

$$C^L(l, r) \simeq \frac{1}{Z} \sum_{j=1}^{N_O} w_j \mathcal{N}_2(\mathbf{p} | \mathbf{p}_j, \sigma_p) \mathcal{N}_1(\theta | \theta_j, \sigma_\theta) \mathcal{N}_1(r | r_j, \sigma_r), \quad (13)$$

where Z is a normalising constant, and $z = (\mathbf{p}, \theta)$.

B. Query density

A query density results from the combination of a contact model with a novel scene O . The scene in this work is represented by a 2D landscape, see Fig. 3a. The purpose of a query density is both to generate and evaluate poses of the corresponding gripper’s link on the new scene (Fig. 3b).

A query density M^{L_i} is a PDF defined as

$$M^{L_i} = P(l, z, v, r), \quad (14)$$

where v denotes a point on a object’s surface in the scene, expressed in the world frame, r is the surface curvature of such a point, z denotes the pose of a link relative to a local frame on the object, and l is the absolute pose in the world frame of the gripper’s link, L_i .

At prediction time, we use a query density M^{L_i} to generate poses of the gripper’s link on the new scene. We achieve this by marginalising with respect to z , v , and r , obtaining the distribution $M^{L_i}(l)$ which models the pose l in the world frame of the link L_i . We approximate the query density by $K_{M^{L_i}}$ kernels centred on the set of weighted robot link poses. As proposed in [7], by sampling from M^{L_i} for each link, a grasp distribution is obtained from which is possible to compute a set of candidate grasps for the new scene. The same learnt contact model allows us to generalise similar surfaces in the novel scene and to adapt the grasp to different sizes of the objects, Fig. 3b.

C. LTV-LQR controller

For each grasp, g , computed by the query density, as described in Sec. III-B, we compute a reach-to-grasp trajectory, $[\mathbf{x}_g(t), \mathbf{u}_g(t)]$, from the current pose of the gripper, $\mathbf{x}_g(0)$. Note that our approach is independent from the method used to generate the trajectory. In our demonstration the trajectories are computed as straight lines to the goal, as shown in Fig. 4. We assumed no obstacles in the free space. However the orientation and aperture of the gripper is computed along the trajectories so that no collision will happen with the skyline of the scene. During a grasp, the gripper closes to generate contacts but penetration is not allowed.

First, the state space in our approach is defined in terms of 2D position and 1D orientation, i.e. $SE(2)$. Without losing generality, we describe a state $\mathbf{x} = [x, y, \theta]^\top$ as a three-element vector, and the control input in terms of velocity, $\mathbf{u} = [\dot{x}, \dot{y}]$. The end-effector orientation, θ , and the pose of the fingers is controlled directly by the system by superimposing the orientation and the finger’s pose planned in the selected trajectory. Using the change in coordinates defined in Eq. 6 we obtain the system formulation in Eq. 8.

The cost-weighted matrices $Q(t)$ and $R(t)$ are designed to drive the user towards the selected trajectory, however their influence need to disappear when the system realises that the user’s intention is to move towards another candidate grasp.

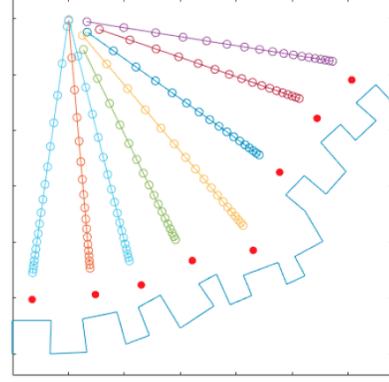


Fig. 4: The red point in this figure represents the candidate grasps, as a 2D pose for the gripper. The system computes a straight line trajectory for each grasp. The trajectory is discretised in a set of waypoints, $\mathbf{x}_g(t)$, in which the local feedback controller in Eq. 8 is centred. Best seen in colours.

This is done by an exponential function as follows

$$Q(t) = \begin{bmatrix} e^{-t} & 0 & 0 \\ 0 & e^{-t} & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (15)$$

$$R(t) = \begin{bmatrix} e^{-t} & 0 \\ 0 & e^{-t} \end{bmatrix} \quad (16)$$

where $Q(0)$ is the cost-weighted matrix associated with the final step in Eq. 9.

Finally we define how the user’s input is integrated in Eq. 10. Typically LTV-LQR computes the difference between the ideal control, \mathbf{u}_0 , at time t and the feedback computed by the matrix K with respect to the state, x , at time t . We extend this formulation to make a prediction about where the user wants to go and how to cope with that. First, the prediction is done by integrating the user’s input, \mathbf{u} , in the state space by

$$\phi(\mathbf{x}, \mathbf{u}, t) = \int_t^{t+1} f(\mathbf{x}(t), \mathbf{u}(t)) dt \quad (17)$$

and then we select the feedback controller with the lowest immediate cost function for the predicted state at time t . Let $\hat{\mathbf{x}} = \phi(\mathbf{x}, \mathbf{u}, t) - \mathbf{x}_g(t)$ be the difference between the predicted state and the desired state for the trajectory associated to the grasp g . Similarly, let $\hat{\mathbf{u}} = \mathbf{u}(t) - \mathbf{u}_g(t)$ be the difference between the user input and the expected input for the trajectory. Then we greedily select the trajectory $[\mathbf{x}_g(t), \mathbf{u}_g(t)]$ such that

$$\arg \min_g \hat{\mathbf{x}}^\top Q(t) \hat{\mathbf{x}} + \hat{\mathbf{u}}^\top R(t) \hat{\mathbf{u}}. \quad (18)$$

IV. EXPERIMENTS

In our experiments we employ a simple two spherical fingers parallel gripper shown in Fig. 2a. We demonstrated to the system a pinch grasp over rectangular shaped object as shown in Fig. 2b. The learnt contact model (Sec. III-A) computes a pdf over the poses for links $L1$ and $L2$

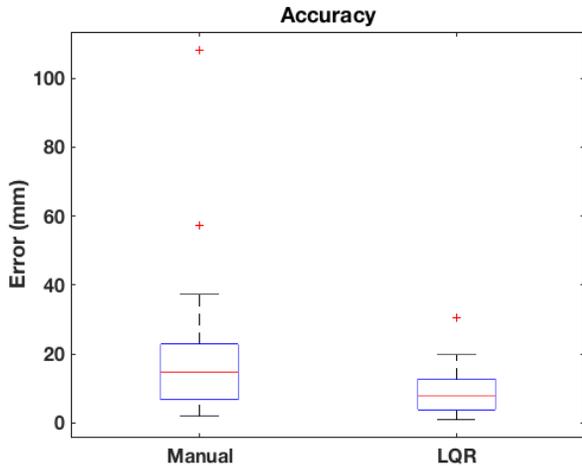


Fig. 5: Position error with respect to the target grasp in mm. Left: the performance of the participants attempting to reach the desired grasp manually. Right: the performance with the help of the LTV-LQR controller. Red lines show the mean error, the bars the standard deviation and the red crosses outliers.

given the object’s surface features. At prediction time, the query density described in Sec. III-B will attempt to sample candidate grasps which have similar characteristics to the learnt contact model.

A. Experimental setup

We collected data from 4 participants: two males and two females between 20 and 40 years of age. All the participants have no history of neuromuscular disorders. Each subject was presented with a random generated scene, as in Fig. 3b, and a random object (i.e. rectangular shape) was selected. We asked the subjects to move the gripper as to grasp the selected object by using the numerical pad of a standard keyboard. The keys 2, 4, 6, 8, were used for down, left, right and up movements respectively and 1, 3, 7 and 9 for the diagonal movements.

We collected a total of 80 trials. Each subject performed 20 trials: 10 trials controlling the gripper manually and 10 trails using the LTV-LQR. We analysed 2 metrics: 1) accuracy in achieving the grasp and 2) execution time form the first control input until the grasp was achieved. Figure 5 shows that our approach is superior to the manual control and allows the user to reach with better accuracy the desired target. In Fig. 6, we show the execution time for all the trials. Our controller allows the user to drive the end-effector towards the target faster, this is because imprecise inputs from the users are filtered out by the controller which still generate a smooth trajectory. However, some reduced performance in time has been registered when the system, especially to the earlier stages of the movement, selects the wrong candidate trajectory. Nonetheless the user is capable to generate a sequence of inputs, i.e. driving the end-effector towards a different candidate, to inform the system about his/her

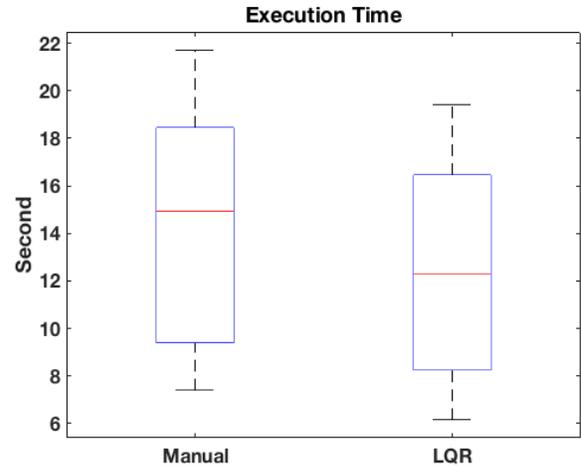


Fig. 6: Execution time for all the participants in seconds. Red lines show the mean error, the bars the standard deviation and the red crosses outliers.

intention which yields to a change in the selected trajectory to support the user.

Although the simplistic scenario, the results encourage the development of AI controllers for semi-autonomous robots. Context-awareness allows the system to understand the scene and generate candidate solutions even in novel scenarios. At the same time, user-awareness interprets the input signals of the user to identify the task, i.e. which object to grasp.

The benefits of our approach are twofold: 1) reduce the complexity of controlling the device, thanks to the ability of the system to infer orientation and fingers’ configurations, and 2) imprecise inputs will be filtered by the controller to obtain a smooth trajectory.

V. CONCLUSION AND FUTURE WORK

The presented approach is a step forward to improve accuracy and usability of semi-autonomous robots such as human operated manipulator for nuclear waste disposal. The key idea is to develop a system that is simultaneously context- and user-aware. We implement the context-awareness by enabling the system to generate a set of candidate grasps on novel scenes, and by planning nominal reach-to-grasp trajectories. For each trajectory a linear time-variant LQR controller is created that identifies the user’s intention from motion commands and guide the user to the intended grasp. The user’s burden is limited to control the end-effector in 2D while the orientation and the pose of the fingers are inferred by the system according to the selected grasp. The system also is capable to recover from wrong decisions when the selected target grasp is not the same as the one chosen by the user.

The results have demonstrated that in our simple scenario the user is capable to reach a desired target faster and with better accuracy than on manual control. Future work will be addressed to extend this type of control to a 3D scenario in which a user controls a real robot arm for tasks like pick

and place, and to investigate whether the user's cognitive burden is reduced by our approach. Additionally, we are also interested to investigate different types of input control, such as EMG signals, to extend this work to benefit patients with neuromuscular disorders and amputees.

REFERENCES

- [1] C. Alvarez Picaza, M. I. Pisarello, and J. E. Monzón. Analysis of the stability control of motors used in biomechanical prostheses. In *VII Latin American Congress on Biomedical Engineering CLAIB 2016*, pp. 561–564. Springer Singapore, 2017.
- [2] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2nd edition, 2000.
- [3] F. Cordella, A. Ciancio, R. Sacchetti, A. Davalli, A. Cutti, E. Guglielmelli, and L. Zollo. Literature review on needs of upper limb prosthesis users. *Frontiers in Neuroscience*, 10:209, 2016.
- [4] Ron Fulbright and Larry M. Stephens. Swami: An autonomous mobile robot for inspection of nuclear waste storage facilities. *Autonomous Robots*, 2(3):225–235, 1995.
- [5] RARC Gopura and Kazuo Kiguchi. Mechanical designs of active upper-limb exoskeleton robots: State-of-the-art and design difficulties. In *Rehabilitation Robotics, 2009. ICORR 2009. IEEE International Conference on*, pp. 178–187. IEEE, 2009.
- [6] Kenichi Kanatani. *Statistical optimization for geometric computation: theory and practice*. Courier Corporation, 2005.
- [7] Marek Kopicki, Renaud Detry, Maxime Adjigble, Rustam Stolkin, Ales Leonardis, and Jeremy L. Wyatt. One-shot learning and generation of dexterous grasps for novel objects. *The International Journal of Robotics Research*, 35(8):959–976, 2015.
- [8] Kent Yee Lui, Hyunjun Cho, ChangSu Ha, and Dongjun Lee. First-person view semi-autonomous teleoperation of cooperative wheeled mobile robots with visuo-haptic feedback. *The International Journal of Robotics Research*, 36(5-7):840–860, 2017.
- [9] Carlos J. Rosales, Federico Spinelli, Claudio Zito, Marco Gabiccini, and Jeremy L. Wyatt. Gpatlasrrt: A tactile exploration strategy for novel object shape modeling. *International Journal of Humanoid Robotics, Special Issue 'Tactile perception for manipulation: new progress and challenges'*, 15(1), 2018.
- [10] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London, 1986.
- [11] M. Spivak. *A Comprehensive Introduction to Differential Geometry*, volume 1. Publish or Perish, 1999.
- [12] Jochen Stüber, Claudio Zito, and Rustam Stolkin. Let's push things forward: A survey on robot pushing. *arXiv preprint arXiv:1905.05138 [cs.RO]* (cs.AI), 2019.
- [13] E. Todorov and W. Li. Embs- iee proceedings of the 25th annual international conference. In *International Conference on Robotics and Automation*, volume 2, pp. 1758–1761. IEEE, 2003.
- [14] Claudio Zito, Marek Kopicki, Rustam Stolkin, Christopher Borst, Florian Schmidt, Maximo A. Roa, and Jeremy L. Wyatt. Sequential re-planning for dextrous grasping under object-pose uncertainty. In *Workshop on Manipulation with Uncertain Models, Robotics: Science and Systems (RSS)*, 2013.
- [15] Claudio Zito, Marek Kopicki, Rustam Stolkin, Christopher Borst, Florian Schmidt, Maximo A. Roa, and Jeremy L. Wyatt. Sequential trajectory re-planning with tactile information gain for dextrous grasping under object-pose uncertainty. In *IEEE Proc. Intelligent Robots and Systems (IROS)*, 2013.
- [16] Claudio Zito, V. Ortienci, M. Adjigble, Marek Kopicki, Rustam Stolkin, and Jeremy L. Wyatt. Hypothesis-based belief planning for dexterous grasping. *arXiv preprint arXiv:1903.05517 [cs.RO]* (cs.AI), 2019.
- [17] Claudio Zito, Rustam Stolkin, Marek Kopicki, Massimiliano Di Luca, and Jeremy L. Wyatt. Sequential re-planning for dextrous grasping under object-pose uncertainty. In *Proc. Workshop on Beyond Robot Grasping: Modern Approaches for Dynamic Manipulation. Intelligent Robots and Systems (IROS)*, 2012.