Visual estimation of articulated objects configuration during manipulation with a humanoid

Antonio Paolillo, Anastasia Bolotnikova, Kevin Chappellet, Abderrahmane Kheddar

Visual estimation of articulated objects configuration during manipulation with a humanoid

Antonio Paolillo, Anastasia Bolotnikova, Kévin Chappellet, Abderrahmane Kheddar

***Abstract*— Robotic manipulation tasks require on-line knowledge of the operated objects' configuration. Thus, we need to estimate online the state of the (articulated) objects that are not equipped with positioning sensors. This estimated state w.r.t the robot control frame is required by our controller to update the model and close the loop. Indeed, in the controller we use the models of the (articulated) objects as additional 'robots' so that it computes the overall 'robots-objects' augmented system's motion and contact interaction forces that fulfill all the limitation constraints together with the physics. Because of the uncertainties due to the floating-base nature of humanoids, we address the problem of estimating the configuration of articulated objects using a virtual visual servoing-based approach. Experimental results obtained with the humanoid robot HRP-4 manipulating the paper drawer of a printer show the effectiveness of the approach.**

I. Introduction

Recently, we have devised a multi-objective task space robot controller formulated as a quadratic program (QP) that includes manipulated objects modeled as an augmentation of the robot structure [1][2]. By doing so, robots and manipulated objects are integrated in a "multi-robot system" that is controlled with a single QP when they come to interact. We have illustrated the use of this multi-robot QP (MQP) control framework in various challenging scenarios in computer graphics animation [1] and robotics [2]. We have also used the MQP to make a humanoid robot drive a car [3]. The driving wheel is (kinematically and dynamically) modeled as a "robot" with a fixed base and one rotational joint. It is then integrated to the MQP to form the multi-robot "humanoid plus driving wheel". The MQP computes desired state acceleration for the overall system that is coherent with both the kinematic constraints of the driving wheel and the contact forces to steer it. In this example, the position of the driving wheel matches exactly that of the rotation of the humanoid's wrist, as the grasping was particularly designed to meet such a request. However, the manipulation task would have been complicated if the robot has to grasp the driving wheel laterally or if it has to re-grasp. In fact, the driving wheel does not have encoders measuring its state.

Doors, drawers, valves, switchers and other articulated tools can also be manipulated by a robot, and modeled as passive robotic structures to be integrated in the MQP. The user models all these object as separate `urdf` files to be loaded by our controller and integrated as a single multi-robot system as well described in [2]. To know the state of the objects, that are not instrumented with sensors, we need to devise external observer to estimate them.

Let us consider the example of a humanoid robot opening the drawer of a piece of furniture (see Fig. 1). The main link of the object is modeled as the base of a robot, and the drawer is considered as a link connected to the base through a prismatic joint. In order to control the opening motion, we can use the position task of our MQP that considers the error between the current position of the drawer $\boldsymbol{p}_{\text{drawer}}(\boldsymbol{q})$, and its desired position. To achieve this task we need the measurement of the vector $\boldsymbol{q}$, i.e. the configuration of the coupled "humanoid-piece of furniture" multi-robot system.

While the configuration of the humanoid is provided by the encoders mounted at each joint, an estimator has to be designed to reconstruct the configuration of the piece of furniture. We could think of using the robot forward kinematics for such estimation, assuming that the drawer is firmly grasped. However, in practice, this does not work for two main reasons: (i) the humanoid is a floating base system and we may have uncertainties to localize it, and (ii) the fingers that pull the printer have no encoders. Furthermore, some humanoid structures (such as HRP-4) have flexibilities in the ankles as well as in the fingers. As a results, we cannot rely much on the pure kinematics of the humanoid to estimate the drawer's joint value. One could think of improving the kinematic measurements integrating other sensors information (such as vision) that would inform about the grasp slipping or the backlash between the robot hand the drawer handle. In a nutshell, one needs to provide the robot with the capability of reconstructing the object configuration as required by the MQP (see Sect. II for the related work).

Through this example, we can extend the reasoning to the necessity of having the reconstruction of articulated objects configuration for the control of this "augmented" system. Knowing the configuration of the objects to be operated, allows to effectively close the loop of the MQP.

This paper addresses this problem:

- formulating (Section III) the estimation of articulated objects configuration as a virtual visual servoing problem (Section IV);
- enabling closed-loop experiments with a humanoid operating the drawer of a printer, using the estimation of the object configuration as feedback in our MQP control framework (Section V).

This work is supported by the EU H2020 COMANOID project (www.comanoid.eu), and by the ROMEO 2 project, (www.projetromeo.com), bpifrance in the framework of the Structuring Projects of Competitiveness Clusters (PSPC)

[1]CNRS-University of Montpellier, Laboratoire d'Informatique de Robotique et de Microéectronique de Montpellier (LIRMM), IDH Group, France.

II. BACKGROUND

Reconstruction of the configuration of articulated objects is a well studied problem in both the computer vision community (tracking of human motion, hands, etc.), and in robotics (tracking of robotic systems).

Several methods to track the motion of the articulated bodies have been proposed over the years. A kinematic tree based parametrization of articulated objects was used in [4]. In this work the tracking was formulated as a tree parameters fitting problem, assuming full geometric model of an object to be known. In [5] the articulated structure tracking is handled as an extension of rigid object tracking. Independent trackers are used to compute motions of every link, then constraints between the links are imposed to find optimal set of motions that satisfy constraints. Multiple hypothesis using derivative- or gradient-free optimization techniques have been studied in the line of works (e.g. [6][7][8]). Such approaches are quite helpful to avoid local minima in optimization. Recently, these methods have also been further studied for estimating force from vision [9][10]. Yet, estimating the configuration of articulated structures with multiple hypothesis methods, can lack frame-to-frame consistency, due to occasional ambiguous observations which cause false hypotheses temporarily to have hight matching scores. This issue is not critical in many applications areas (surveillance, computer graphics, etc.), but in robotic closed-loop control schemes, such estimation inconsistency could result in a bad behavior, such as jumps and jerky motions.

The motion of articulated objects can also be estimated using depth information in a GPU-based implementation of an Extended Kalman Filter (EKF) in [11]. This method has been extended further to consider physical constraints in tracker objective function by using contact information in robotic object manipulation scenario [12]. Another depth based method for estimating the state of a robotic arm was proposed in [13]. It demonstrated robustness to calibration errors in a closed-loop manipulation task. Combination of depth and joint encoders data was used in [14] to track the state of robotic arm. For some robotic platforms, especially humanoids, end-effector distance from the camera may not exceed the minimum distance for depth data acquisition by a standard range sensor. In such case, RGB data processing is the only reliable source of visual information. Articulation tracking can also be done using images collected by a multi-camera system and then processed by a particle filter [15]. These methods are yet computationally expensive to be applied at the low-level robot control.

Model-based approaches using monocular cameras are interesting techniques to achieve fast and accurate estimate of articulated objects configuration. In [16] a Kalman filter-based tracking, using multiple models (such as the geometric and the appearance model of the object), is proposed to recover values of joint position and velocity, but not that of the floating base. Another method [17] uses a virtual visual servoing-based approach [18] providing the configuration of the object, but not expressed with the classical generalized coordinates. Further computations should be added to retrieve the joint variables.

An advantage of the model-based approaches is that the object and the visual features trackers can work together cooperatively: knowing the model of the object, the tracking of features leads to the reconstruction of the object configuration and *vice-versa*. Extending this concept, the features motion can also reveal geometric information of the observed object, that in turn is used to better track the features. This idea is exploited in [19], that estimates the kinematic structure of the observed object combining the manipulation task with the perception algorithm. With the same principle, color and depth (RGB-D) information are processed by an EKF to provide also a measurement of the joint values in [20]. In [21], RGB-D data is processed in a unified framework able to estimate the pose, the shape and the structure of the observed object. These approaches do not need the model of the object, being itself estimated, and have been validated with simple articulated objects. Furthermore, they are not guaranteed to converge fast or to be reliable in all circumstances. Some are computationally expensive.

In our attempt to find the articulation tracking solution, whose formalism can be defined and used as a part of MQP in the closed-loop object manipulation control, no existing method could fit our requirements. Therefore, we took inspiration from previous works to devise a tracker whose formalism suits the MQP requirements and is presented in the remaining of the paper.

III. PROBLEM FORMULATION

To track articulated objects for robotic manipulation, we propose a method based on the so-called virtual visual servoing. To be self-contained, we briefly recall the basics of this technique and formulate our tracking problem.

Visual Servoing (VS) achieves a cartesian task using visual feedback [22]: the control provides the camera velocity $\boldsymbol{v}_c$ in order to zero the error between the measured and the desired value of visual features, denoted with $\boldsymbol{s}$ and $\boldsymbol{s}^*$, respectively.

VS can be exploited in a dual way: let a *virtual camera* moving in the cartesian space, whose unknown pose $\boldsymbol{p}$ is defined w.r.t. an observed object, i.e. in correspondence of some *virtual visual features*, collected in the vector $\boldsymbol{s}(\boldsymbol{p})$[1]. The real pose of the camera $\boldsymbol{p}^*$ is defined in correspondence of some measured visual features, collected in $\boldsymbol{s}^*$. The convergence of $\boldsymbol{s}(\boldsymbol{p})$ to $\boldsymbol{s}^*$ implies the convergence of $\boldsymbol{p}$ to $\boldsymbol{p}^*$. Thus, in this context, the VS control law is used as "estimator" of the camera pose[2]. This methodology, known as Virtual Visual Servoing (VVS), was introduced in the framework of augmented reality [23] [18]. The same technique can be used to estimate the pose of the object moving in the camera scene, and extended to articulated object tracking as in [17] that inspired our work. In particular, we share with [17] the

[1]Note that $\boldsymbol{s}(\boldsymbol{p})$ is reconstructed by using the projection model of the camera. Thus, to be rigorous, it is dependent also on the camera intrinsic parameters, assumed to be known, that here are omitted to highlight the dependence on the unknown variable $\boldsymbol{p}$.

[2]For example, the camera pose can be reconstructed using $\boldsymbol{p} = \exp(\boldsymbol{v}_c)$.

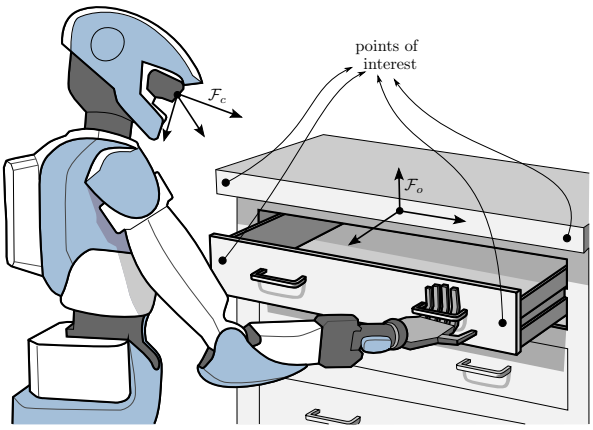


Fig. 1. A humanoid robot manipulating the drawer of a furniture.

rationale of the approach, but we use a lighter formalism allowing an easier implementation of the method.

Let us consider the case of a camera looking at an articulated object, as in the depicted scenario of Fig. 1, where a humanoid robot is opening the drawer of a piece of furniture. The camera frame of the robot $\mathcal{F}_c$ is used as reference in the algorithm. Its origin is placed at the focal point and the z-axis is aligned with the focal axis; the y-axis points downwards and the x-axis completes the right-handed coordinates system. Another frame of interest, $\mathcal{F}_o$, is arbitrarily defined at the floating base (FB) of the object.

The *real* configuration of the articulated object (the piece of furniture in the presented example) is denoted by

$$\boldsymbol{q}^* = \begin{pmatrix} \boldsymbol{p}_o \\ \boldsymbol{\sigma}_o \\ \boldsymbol{q}_j \end{pmatrix} \in \mathbb{R}^{7+n} \tag{1}$$

where $\boldsymbol{p}_o \in \mathbb{R}^3$ and $\boldsymbol{\sigma}_o \in \mathbb{S}^3 \subset \mathbb{R}^4$ are the position and orientation (expressed with a quaternion) of $\mathcal{F}_o$ w.r.t. $\mathcal{F}_c$, respectively. Quaternions prevent the orientation representation from being singular. The vector $\boldsymbol{q}_j \in \mathbb{R}^n$ in (1) is the vector of the generalized coordinates describing the internal configuration of the object, composed of n joints.

Let us define a vector $\boldsymbol{\rho} \in \mathbb{R}^{3p}$ of p points of interest (PoI) on the articulated object, whose projection on the image plane of the camera provides p corresponding visual features.

The real motion of the articulated object (1) produces the motion of *real visual features* collected in the vector $\boldsymbol{s}^* \in \mathbb{R}^{2p}$. These features are actually observable and measurable on the image plane of the camera.

Whereas, the *virtual* configuration of the articulated object, denoted with $\boldsymbol{q}$, affects the motion of *virtual visual features*, gathered in the vector $\boldsymbol{s}(\boldsymbol{q}) \in \mathbb{R}^{2p}$. Note that the virtual visual features depend on (i) the geometric model of the object, assumed to be known and used to calculate the location of the virtual PoI in $\mathcal{F}_c$, and (ii) the camera intrinsic parameters, used to project the virtual PoI on the image plane. Since $\mathcal{F}_c$ is the reference frame, we do not need the camera extrinsic parameters to build the projection model.

Assuming to known $\boldsymbol{\rho}$ and measuring $\boldsymbol{s}^*$, the aim of the tracking is to estimate $\boldsymbol{q}^*$. Only rigid objects are considered.

IV. VVS-Based Tracking of Articulated Objects

The approach used to address the articulated object tracking problem relies on the VVS paradigm, i.e. uses a visual controller to estimate the derivative of the configuration vector. Thus, the estimate $\boldsymbol{q}$ is reconstructed through integration.

The error of the proposed control scheme is the difference between $\boldsymbol{s}(\boldsymbol{q})$ and $\boldsymbol{s}^*$ defined in the previous section, i.e., $\boldsymbol{e} = \boldsymbol{s}(\boldsymbol{q}) - \boldsymbol{s}^*$. The dynamics of this error says how the visual features change over time, and can be written as follows:

$$\dot{\boldsymbol{e}} = \dot{\boldsymbol{s}}(\boldsymbol{q}) - \dot{\boldsymbol{s}}^* \tag{2}$$

where the dot over the variables denotes the time derivative.

The motion of the virtual visual features is a consequence of the motion of the virtual object (i.e. its PoI) w.r.t. the camera. In $\mathcal{F}_c$, the 6D velocity of each PoI can be expressed using the geometry of the object, i.e., $\boldsymbol{v}_i = \boldsymbol{J}_i(\boldsymbol{q})\dot{\boldsymbol{q}}$ where $\boldsymbol{J}_i$ is the $6 \times (7+n)$ Jacobian of the i-th PoI, $i = 1, \ldots, p$. Thus, the dynamics of each virtual visual feature can be written as

$$\dot{\boldsymbol{s}}_i(\boldsymbol{q}) = -\boldsymbol{L}_i \boldsymbol{J}_i(\boldsymbol{q})\dot{\boldsymbol{q}} = \boldsymbol{A}_i \dot{\boldsymbol{q}} \tag{3}$$

where $\boldsymbol{L}_i$ is the image Jacobian associated to the i-th virtual visual feature [22]; it depends on the depth of the corresponding PoI, that is available in the estimation process. The minus in (3) explains that the apparent motion of the features $\boldsymbol{s}(\boldsymbol{q})$ is due to the motion of the observed object (actually, its PoI) w.r.t. the camera (typical of *hand-to-eye* systems [24]). The dynamics of the error can be now explicitly written:

$$\dot{\boldsymbol{e}} = \boldsymbol{A}\dot{\boldsymbol{q}} - \dot{\boldsymbol{s}}^* \tag{4}$$

where we define, following the nomenclature in [17], $\boldsymbol{A} = (\boldsymbol{A}_1, \ldots, \boldsymbol{A}_p)^T$ as the $2p \times (7+n)$ *articulation matrix*, that relates the FB and joints velocity of the articulated object to the velocity of the visual features. Imposing a stable dynamics of the error ($\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}, \lambda > 0$), we derive the following "control law"

$$\dot{\boldsymbol{q}} = -\lambda \underline{\boldsymbol{A}}^{\#} \boldsymbol{e} + \underline{\boldsymbol{A}}^{\#} \underline{\dot{\boldsymbol{s}}}^* \tag{5}$$

where the term depending on $\underline{\dot{\boldsymbol{s}}}^*$ introduces an anticipatory action, improving the tracker performance. The bars below the variables denote approximations. The controller's local stability is ensured if $\boldsymbol{A}$ and its approximation are full-rank, and we use a good approximation of $\boldsymbol{A}$ and $\dot{\boldsymbol{s}}^*$. If $(n+7) > p$ (as in the presented results), it has to be ensured that the configuration of the object is not singular [24].

Note that we refer to (5) as a control law, being the term borrowed from the VS nomenclature. However, it actually is an *estimator* providing, at steady state, an estimate of the object's FB and joint velocities $\dot{\boldsymbol{q}}_k$ at each time kT_s (T_s being the sampling time of the algorithm and k the loop iterator). From this, $\boldsymbol{q}_k$, an estimate of the real object configuration, is obtained by numerical integration. To avoid the error introduced by brute-force normalization-based methods, the derivative of the quaternion is integrated using a closed-form exponential map method [25]. The other elements of $\boldsymbol{q}_k$ are

Algorithm 1 VVS-based tracking of articulated objects

for each new image frame $\boldsymbol{I}$ **do**
  $\boldsymbol{s}^* \leftarrow$ DETECT FEATURES($\boldsymbol{I}$)
  $\boldsymbol{\rho} \leftarrow$ UPDATE MODEL($\boldsymbol{q}$)
  for each visual feature i **do**
    $\boldsymbol{J}_i \leftarrow$ COMPUTE JACOBIAN($\boldsymbol{q}, \boldsymbol{\rho}_i$)
    $\boldsymbol{s}_i(\boldsymbol{q}) \leftarrow$ PROJECT($\boldsymbol{\rho}_i$)
    $\boldsymbol{L}_i \leftarrow$ COMPUTE IMAGE JACOBIAN($\rho_{i,z}, \boldsymbol{s}_i(\boldsymbol{q})$)
    $\boldsymbol{A}_i = -\boldsymbol{L}_i \boldsymbol{J}_i$
  end for
  $\dot{\boldsymbol{q}} = -\lambda \boldsymbol{A}^{\#} \boldsymbol{e} + \boldsymbol{A}^{\#} \dot{\boldsymbol{s}}^*$
  $\boldsymbol{q} \leftarrow$ INTEGRATION($\dot{\boldsymbol{q}}$)
end for

obtained with Euler explicit integration:

$$\boldsymbol{q}_k = \begin{pmatrix} \boldsymbol{p}_{o,k-1} + (\dot{\boldsymbol{p}}_{o,k} + \dot{\boldsymbol{p}}_{o,k-1}) \frac{T_s}{2} \\ \exp(\boldsymbol{\Omega}_{k-1} T_s) \boldsymbol{\sigma}_{o,k-1} \\ \boldsymbol{q}_{j,k-1} + (\dot{\boldsymbol{q}}_{j,k} + \dot{\boldsymbol{q}}_{j,k-1}) \frac{T_s}{2} \end{pmatrix} \tag{6}$$

where $\boldsymbol{\Omega}$ is the 4×4 skew matrix of the FB angular velocity.

Algorithm 1 presents the complete procedure to compute the vector $\boldsymbol{q}$. Each new image is processed to detect the visual features filling $\boldsymbol{s}^*$. Using the current estimation $\boldsymbol{q}$, the model of the object is updated, so that the positions of the PoI $\boldsymbol{\rho}$ are also estimated and available for the subsequent computations. Then, for each visual features $i = 1, \ldots, p$ these steps are performed:

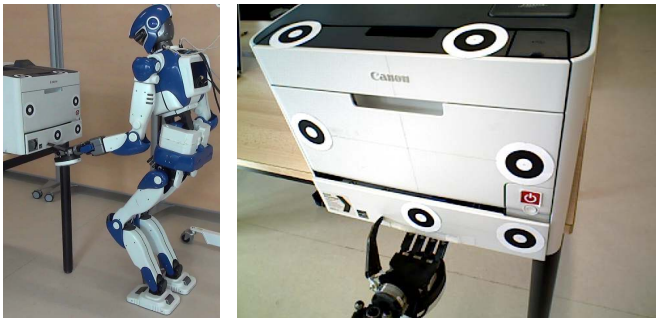
- the Jacobian $\boldsymbol{J}_i$ of the corresponding PoI is computed using the current $\boldsymbol{q}$ and the estimated $\boldsymbol{\rho}_i$;
- the virtual visual feature $\boldsymbol{s}_i(\boldsymbol{q})$ is obtained projecting $\boldsymbol{\rho}_i$ on the image plane;
- the interaction matrix $\boldsymbol{L}_i$ is computed using the estimated depth of the point (i.e., the z-coordinate of the point $\boldsymbol{\rho}_i$) and the coordinates of the visual feature;
- finally, the articulation matrix $\boldsymbol{A}_i$ is obtained.

Once these operations are repeated for all the features, the articulation matrix is fully composed. Finally, $\dot{\boldsymbol{q}}$ is computed and $\boldsymbol{q}$ is obtained by numerical integration.

The method can reconstruct also other information about the object such as the length of the links, the PoI position in the presented study or even the composition of the CAD model, that is assumed to be known. This can be achieved modeling the distances as *virtual* prismatic joints. We exemplify this functionality in the next section.

V. EXPERIMENTAL RESULTS

For our experiments we used the humanoid robot HRP-4, that is equipped with a Xtion PRO LIVE RGB-D sensor. The Xtion is used as monocular camera, providing images with a resolution of 640×480 pixels at 30 Hz. A calibration procedure provided the intrinsic parameters used in the projection model of the algorithm. The images are processed by WhyCon [26], a vision-based localization library that detects proper markers placed in the field-of-view of the camera, and also gives an estimate of their position in the camera frame. In particular, p WhyCon markers are placed at known positions on the articulated object to track. These markers represent the p PoI in our algorithm, and the detection of their corresponding visual features fills the vector $\boldsymbol{s}^*$. For each new set of detected visual features, the algorithm described in Section IV provides an estimate of the articulated object configuration. The implementation of the algorithm is based on the Robotics and Vision Control toolbox [27], while the communication and the control of the robot have been managed by using the ROS framework.



(a) (b)

Fig. 2. Experimental setup: (a) HRP-4 manipulating the paper drawer of a printer and (b) the corresponding image acquired by the onboard camera.

The approach has been used to make HRP-4 manipulate the paper drawer of a printer (Fig. 2a). The printer has been structured with six WhyCon markers, placed at known positions, four on the FB and two on the drawer, as shown in Fig. 2b. The configuration vector $\boldsymbol{q} \in \mathbb{R}^9$ that we provide is composed of (i) the pose (position vector and quaternion) of the printer FB, (ii) the value of the prismatic joint of the drawer, q_1, and (iii) the distance between the two markers on the drawer, q_2, modeled as a virtual prismatic joint. The configuration vector has been initialized to $\boldsymbol{q}_0 = (0.0, 0.1, 0.6, 1.0, 0.0, 0.0, 0.0, 0.0, 0.1)^T$. To have a smooth transient phase and good tracking performance, we designed a profile of the gain λ dependent on the VVS error (high when the norm of the error is low and *vice versa*). The maximum and minimum values of the gain were set to $\lambda_{\max} = 5$ and $\lambda_{\min} = 2$, respectively. Since the computation of the visual features derivative was noisy and the camera did not move excessively during the execution of the experiment, we disabled the derivative action in (5).

To validate our approach, we compare the results of the proposed VVS-based tracking algorithm with a Singular Value Decomposition (SVD) based method for rigid motion reconstruction. It uses two sets of points: m PoI on the FB expressed in camera frame, ${}^c\boldsymbol{\rho}$, and the same points expressed in the object frame, ${}^o\boldsymbol{\rho}$. The first set is provided by WhyCon, the latter is known, manually measured. The reconstruction of the object pose (position vector $\boldsymbol{p}_o$ and rotation matrix $\boldsymbol{R}_o$) is formulated as a least squares error problem:

$$(\boldsymbol{R}_o, \boldsymbol{p}_o) = \underset{\boldsymbol{R}_o, \boldsymbol{p}_o}{\arg\min} \sum_{i}^{m} \| (\boldsymbol{R}_o \, {}^o\boldsymbol{\rho}_i + \boldsymbol{p}_o) - {}^c\boldsymbol{\rho}_i \|^2. \tag{7}$$

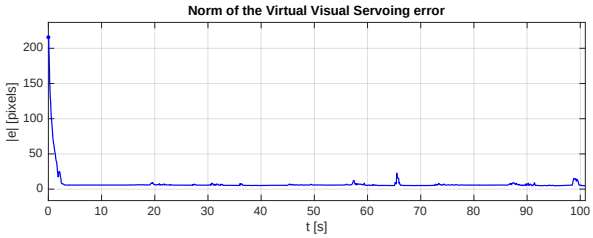


Fig. 3. Experimental results: norm of the VVS error.

To find an optimal combination of $\boldsymbol{R}_o$ and $\boldsymbol{p}_o$ that satisfies the minimization problem in (7), we apply an SVD on the cross-covariance matrix of the two points distributions ($^c\boldsymbol{\rho}$ and $^o\boldsymbol{\rho}$), that results in the decomposition $\boldsymbol{U\Sigma V}^T$. The sum in (7) is minimized when $\boldsymbol{R}_o = \boldsymbol{VU}^T$. Then, the translation is computed as $\boldsymbol{p}_o = \boldsymbol{\mu_c} - \boldsymbol{R}_o\boldsymbol{\mu}_o$, where $\boldsymbol{\mu}_c$ and $\boldsymbol{\mu}_o$ are the centroids of the sets $^c\boldsymbol{\rho}$ and $^o\boldsymbol{\rho}$, respectively. For the validation of the drawer prismatic joint, we consider the 3D position of a marker on the drawer in $\mathcal{F}_o$ reconstructed using $\boldsymbol{R}_o$ and $\boldsymbol{p}_o$. The amount of drawer opening is equal to the position of the drawer marker on z-axis of $\mathcal{F}_o$ (Fig. 1). For the validation of estimated distance between two markers on the drawer, Euclidean distance between corresponding points is computed using coordinates in $\mathcal{F}_c$ provided by WhyCon.

As discussed in the Sect. I, the robot is controlled with our MQP [2]. Indeed, in order to achieve the closed-loop behavior, it is possible to define the error between the current value of the drawer's joint and a desired joint target: $\tau_q = (q_1 - q_{1,d})$. Here, $q_{1,d}$ is specified by the user (it could be defined by a higher level planning), while q_1 is provided by our method, see Sect. IV. This term is actually added to the cost function of the MQP after the robot grasps the drawer

$$w_q \left\| \ddot{\tau}_q + 2\sqrt{k_p}\dot{\tau}_q + k_p\tau_q \right\|, \tag{8}$$

where k_p is a positive gain and $\ddot{\tau}_q = \boldsymbol{J}_q\ddot{q}_1 + \dot{\boldsymbol{J}}_q\dot{q}_1$; $\boldsymbol{J}_q$ is the Jacobian of the task and w_q a given weight.

The experiment starts with the robot already at the operational configuration, at stand position, with the left hand grasping the drawer of the printer. During the experiment, a user sends opening/closing commands to the MQP.

Figure 3 shows the norm of the VVS error. After a transient time required to make the virtual visual features converge on their real counterparts, the error decreases exponentially and remains below a threshold.

The position of the FB as estimated by the VVS (blue continuous lines with triangle markers) and by the SVD-based method (red dashed line) is shown in the plots of Fig. 4. Again, after an initial transient required to recover the bad initialization, the signals provided by the VVS converge to the position of the object; the curves provided by the SVD-based method validate the estimation results. Similarly, plots of the FB orientation (transformed in roll-pitch-yaw angles) are presented in Fig. 5. The effectiveness of the VVS at estimating the pose of the FB is validated by the comparison with SVD-based method. Furthermore, it appears to be less noisy and more appropriate to be used as control feedback.

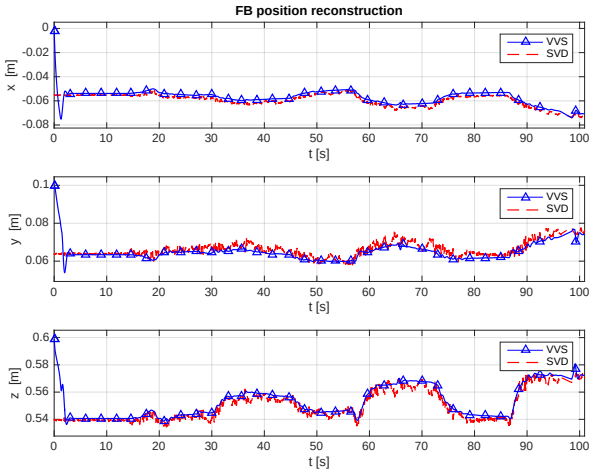


Fig. 4. Experimental results: position of the object w.r.t. the camera frame. From top to bottom: x, y and z-coordinate.

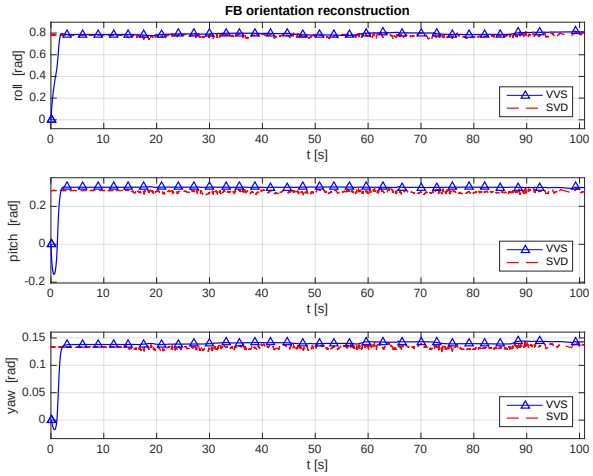


Fig. 5. Experimental results: orientation of the object w.r.t. the camera frame. From top to bottom: roll, pitch and yaw angle.

The plots in Fig. 6 refer to the joint of the printer drawer. The black dash-dot line shows the desired command sent to the MQP, that is followed by the estimation of the joint position, as shown by the curves. The control has been enabled around time 15 s. A phase can be observed in tracking. This is due to two main factors: the backlash in the humanoid-drawer system, especially when there are two consecutive opening/closing commands, and the task error decrease rate of the MQP that controls the robot. Note also that the estimation algorithm computes an output for each set of detected features. It could happen that the detection process fails creating instant lack of estimation.

Finally, the plot in Fig. 7 shows the effectiveness in estimating the distance between the markers placed on the printer drawer, modeled as a prismatic joint in our algorithm.

The experiment is also shown in the video available at `https://youtu.be/Vr2LUEovof8`.

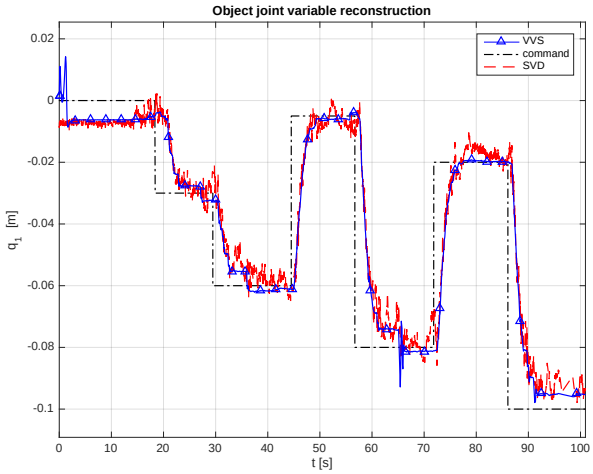


Fig. 6. Experimental results: position of the drawer prismatic joint.

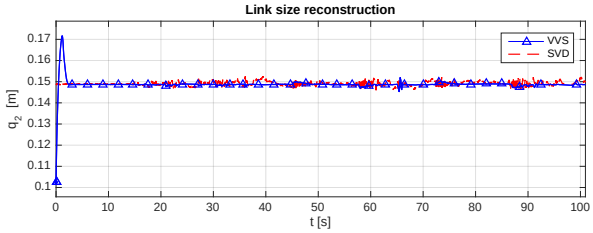


Fig. 7. Experimental results: distance between the markers of the drawer.

VI. Conclusions and Future Work

In this paper we addressed the problem of estimating the configuration of an articulated object to be manipulated by a humanoid robot. The approach is based on the virtual visual servoing paradigm, and uses visual information coming from the robot onboard camera to reconstruct the pose of the object floating base and its joints configuration. The output of the estimator is fed back to a multi-robot quadratic program framework for controlling the humanoid HRP-4. Experimental results showed the effectiveness of the approach in manipulating the paper drawer of a printer.

Future work will investigate the use of lines instead of points as features, to avoid the structuring the object with known markers. Robotic structures and more complex articulated objects can also be tracked with this method. Furthermore, future perspectives can be traced in the field of safe physical human-robot interaction, where the detection and prediction of a human partner motion is very important. In fact, the proposed algorithm can be extended to the tracking of human bodies.

References

[1] J. Vaillant, K. Bouyarmane, and A. Kheddar, "Multi-character physical and behavioural interactions controller," *IEEE Transactions on Visualization and Computer Graphics*, 2017.

[2] K. Bouyarmane, J. Vaillant, K. Chappellet, and A. Kheddar, "Multi-robot and task-space force control with quadratic programming," *IEEE Transactions on Robotics*, submitted.

[3] A. Paolillo, P. Gergondet, A. Cherubini, M. Vendittelli, and A. Kheddar, "Autonomous car driving by a humanoid robot," *Journal of Field Robotics*, 2017.

[4] D. G. Lowe, "Fitting parameterized three-dimensional models to images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 5, pp. 441–450, 1991.

[5] T. Drummond and R. Cipolla, "Real-time visual tracking of complex structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 932–946, 2002.

[6] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Efficient model-based 3D tracking of hand articulations using kinect," in *British Machine Vision Conference*, 2011, pp. 1–11.

[7] ——, "Tracking the articulated motion of two strongly interacting hands," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1862–1869.

[8] G. Park, A. Argyros, and W. Woo, "Efficient 3D hand tracking in articulation subspaces for the manipulation of virtual objects," in *Computer Graphics International*, 2016, pp. 33–36.

[9] T.-H. Pham, A. Kheddar, A. Qammaz, and A. A. Argyros, "Towards force sensing from vision: Observing hand-object interactions to infer manipulation forces," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2810–2819.

[10] T.-H. Pham, N. Kyriazis, A. Argyros, and A. Kheddar, "Hand-object contact force estimation from markerless visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[11] T. Schmidt, R. Newcombe, and D. Fox, "DART: Dense articulated real-time tracking," in *Robotics: Science and Systems*, 2014.

[12] T. Schmidt, K. Hertkorn, R. Newcombe, Z. Marton, M. Suppa, and D. Fox, "Depth-based tracking with physical constraints for robot manipulation," in *IEEE Int. Conf. on Robotics and Automation*, 2015, pp. 119–126.

[13] M. Klingensmith, T. Galluzzo, C. Dellin, M. Kazemi, J. A. D. Bagnell, and N. Pollard , "Closed-loop servoing using real-time markerless arm tracking," in *ICRA workshop on Humanoids*, 2013.

[14] M. Krainin, P. Henry, X. Ren, and D. Fox, "Manipulator and object tracking for in-hand 3D object modeling," *The International Journal of Robotics Research*, vol. 30, no. 11, pp. 1311–1327, 2011.

[15] J. Deutscher and I. Reid, "Articulated body motion capture by stochastic search," *International Journal of Computer Vision*, vol. 61, no. 2, pp. 185–205, 2005.

[16] K. Nickels and S. Hutchinson, "Model-based tracking of complex articulated objects," *IEEE Trans. on Robotics and Automation*, vol. 17, no. 1, pp. 28–36, 2001.

[17] A. I. Comport, E. Marchand, and F. Chaumette, "Kinematic sets for real-time robust articulated object tracking," *Image and Vision Computing*, vol. 25, no. 3, pp. 374–391, 2007.

[18] A. I. Comport, E. Marchand, M. Pressigout, and F. Chaumette, "Real-time markerless tracking for augmented reality: the virtual visual servoing framework," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 4, 2006, pp. 615–628.

[19] D. Katz and O. Brock, "Manipulating articulated objects with interactive perception," in *IEEE Int. Conf. on Robotics and Automation*, 2008, pp. 272–277.

[20] R. Martín Martín and O. Brock, "Online interactive perception of articulated objects with multi-level recursive estimation based on task-specific priors," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2014, pp. 2494–2501.

[21] R. Martín Martín, S. Höfer, and O. Brock, "An integrated approach to visual perception of articulated objects," in *IEEE Int. Conf. on Robotics and Automation*, 2016, pp. 5091–5097.

[22] F. Chaumette and S. Hutchinson, "Visual Servo Control, Part I: Basic Approaches," *IEEE Robotics & Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.

[23] E. Marchand and F. Chaumette, "Virtual Visual Servoing: a framework for real-time augmented reality," *Computer Graphics Forum*, vol. 21, no. 3, pp. 289–297, 2002.

[24] F. Chaumette and S. Hutchinson, "Visual Servo Control, Part II: Advances Approaches," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 109–118, 2007.

[25] F. Zhao and B. G. M. van Wachem, "A novel quaternion integration approach for describing the behaviour of non-spherical particles," *Acta Mechanica*, vol. 224, no. 12, pp. 3091–3109, 2013.

[26] T. Krajník, M. Nitsche, J. Faigl, P. Vaněk, M. Saska, L. Přeučil, T. Duckett, and M. Mejail, "A practical multirobot localization system," *Journal of Intelligent & Robotic Systems*, vol. 76, no. 3, pp. 539–562, 2014.

[27] P. I. Corke, *Robotics, vision and control*. Springer, 2011.