

Visually Guided Model Predictive Robot Control via 6D Object Pose Localization and Tracking

Mederic Fourmy[♣] Vojtech Priban[♣] Jan Kristof Behrens[♣] Nicolas Mansard[◇] Josef Sivic[♣] Vladimir Petrik[♣]

Abstract—The objective of this work is to enable manipulation tasks with respect to the 6D pose of a dynamically moving object using a camera mounted on a robot. Examples include maintaining a constant relative 6D pose of the robot arm with respect to the object, grasping the dynamically moving object, or co-manipulating the object together with a human. Fast and accurate 6D pose estimation is crucial to achieve smooth and stable robot control in such situations. The contributions of this work are three fold. First, we propose a new visual perception module that asynchronously combines accurate learning-based 6D object pose localizer and a high-rate model-based 6D pose tracker. The outcome is a low-latency accurate and temporally consistent 6D object pose estimation from the input video stream at up to 120 Hz. Second, we develop a visually guided robot arm controller that combines the new visual perception module with a torque-based model predictive control algorithm. Asynchronous combination of the visual and robot proprioception signals at their corresponding frequencies results in stable and robust 6D object pose guided robot arm control. Third, we experimentally validate the proposed approach on a challenging 6D pose estimation benchmark and demonstrate 6D object pose-guided control with dynamically moving objects on a real 7 DoF Franka Emika Panda robot.

I. INTRODUCTION

Visually-guided control is at the core of many robotic applications, from path following by mobile robots [1] to visual servoing [2]. In order to achieve a stable and robust feedback loop, the perception system has to recover the estimated state both accurately and at a high rate. In the context of object manipulation, a commonly chosen state representation is the 6D pose of objects of interest, *i.e.*, the 3D translation and 3D rotation of the objects in the scene with respect to the camera coordinate frame. While some manipulation tasks can be achieved with a static scene model [3], many applications are of an inherently dynamic nature with human-robot handovers [4], [5], human-robot co-manipulation [6] or mobile manipulation [7] being the prime examples. This is challenging as it requires accurate and low-latency 6D pose estimation of the target objects in the scene. In addition, pose estimates need to be integrated with a robust and reactive controller that is capable of meeting the dynamic requirements of the application.

[♣] CIIRC, Czech Technical University in Prague

[◇] LAAS-CNRS, Université de Toulouse, CNRS, Toulouse

This work was partly supported by the AGIMUS project, funded by the European Union under GA no.101070165, by the European Regional Development Fund under project Robotics for Industry 4.0 (reg. no. CZ.02.1.01/0.0/0.0/15_003/0000470), and by the Czech Science Foundation (project no. GA21-31000S). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.

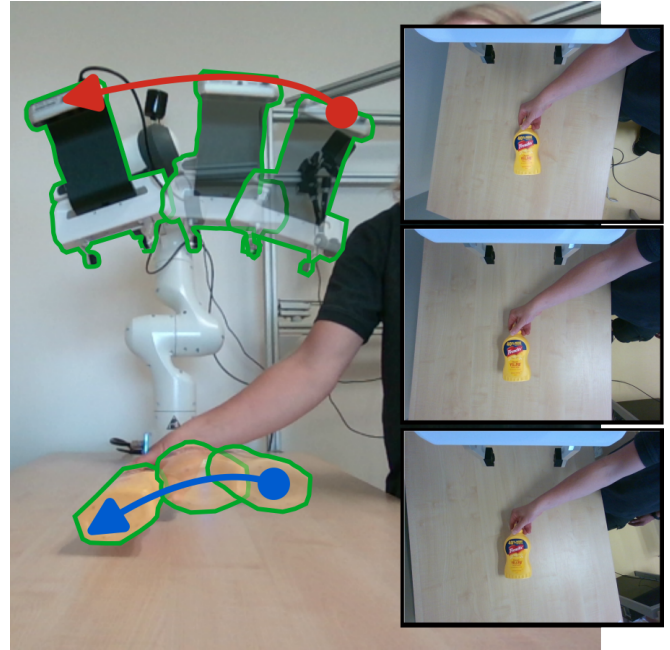


Fig. 1: **Robot arm control by 6D pose of the object.** The objective is to control the robot arm with a mounted camera (red arrow) by commanding joint torques such that the object 6D pose (blue arrow) w.r.t. the camera remains constant. This is illustrated by three frames (see insets) captured by the robot camera corresponding to the robot/object poses shown by green contours in the main image. Please note (see the insets) how the object pose remains stable while the background changes in the captured frames. **More results and experimental analysis in the companion video.**

Despite promising recent progress in object detection and 6D pose estimation [8], [9], [10], [11], [12], [13], 6D object pose estimation algorithms usually focus on accuracy rather than speed. On the other end, object 6D pose tracking methods offer fast pose updates of already detected objects, but require an initialization from the user [14]. As a result of these limitations, many real-world applications rely instead on fiducial markers [15], [16], [17], motion capture systems [6], [5] or ad-hoc detection such as color based segmentation [18].

In this paper, we propose a visual perception module that builds on (i) state-of-the-art accurate learning-based 6D object pose detector and (ii) state-of-the-art high-rate model-based 6D pose tracker to achieve object pose estimation limited only by the rate of image acquisition. Further, we develop a visually guided robot controller based on the

model predictive control (MPC) that is able to reactively incorporate perception updates to meet application targets (e.g., positioning the robot’s gripper). The *anticipatory* nature of MPC makes it particularly suitable for generating efficient motions in real time [19]. In principle, the only input data required is a 3D object model to configure the 6D pose tracker and the 6D pose estimator. Our method runs the pose detector and several instances of the object tracker in separate processes and uses the results of the pose detector to re-initialize the trackers. New images are directly fed to a tracker so that a pose estimation result is available within the tracker runtime of approximately 5 ms. To quantitatively validate our perception module, we build on the BOP challenge evaluation [9] and the YCBV dataset [20] to measure the performance of the proposed method and several baseline methods. Lastly, we demonstrate visually-guided robot arm control with hand held objects (see Fig. 1).

Contributions. The paper has the following three main contributions: (i) we propose a new visual perception module that asynchronously combines accurate learning-based 6D object pose localizer and a high-rate model-based 6D pose tracker; (ii) we develop a visually guided robot arm controller that leverages the new visual perception module in a torque-based model predictive control algorithm; and (iii) we experimentally validate the proposed approach on a challenging 6D pose estimation benchmark and demonstrate 6D object pose-guided control with dynamically moving objects on a real 7 DoF Franka Emika Panda robotic platform. We will make the code publicly available.

II. RELATED WORK

Object 6D localization. The field of object detection and 6D pose estimation has shown impressive progress over recent years, which has been documented and fostered by the *benchmark for 6D object pose estimation* (BOP) and the associated BOP challenge [8], [9]. Most methods follow a two-step approach, first performing object detection in RGB frames [21] followed by 6D pose estimation, assuming the availability of object meshes. Learning-based techniques have dominated the field. Among the diverse approaches, render-and-compare methods [22], [10], [13], have achieved superior performance by iteratively refining the 6D pose based on predictions by a neural network. Due to their iterative nature these methods achieve superior performance but are slow to be used in real-time control. In this work, we propose to combine a slow ”render and compare” 6D pose localizer with a fast 6D pose tracker. Although the proposed approach can work with an arbitrary localizer, we use pre-trained CosyPose [10] in all our experiments.

Object 6D pose tracking. When a good initial guess of the object 6D pose is available, it can be tracked frame to frame by fast local methods. Object pose tracking methods rely on object edges [23], extracted point features [24], [25], or depth [26]. Region-based tracking approaches propose to solve the 2D object shape segmentation and 6D pose tracking problems jointly by constructing an image-wise posterior distribution [27]. Although initial versions required

highly optimized GPU implementations to run at the camera frequency [28], [29], a sparse formulation based on contour point sampling [30] dramatically reduces the computation time, down to a few milliseconds per image [31], [14] on a single CPU. In this work, we rely on the ICG implementation [14] that features both region-based and depth modalities. The combination of the 6D pose localizer and the local 6D pose tracker that we propose in this paper combines the benefits of the both world, *i.e.* the detection capability and the accuracy of the localizer and speed of the tracker.

Visual servoing. Visual servoing aims at building a closed-loop controller using visual information from a camera stream to achieve a certain goal. The various methods are traditionally classified into two broad categories [2]: (i) image-based visual servoing [32], [33], which uses 2D geometric primitives (e.g., points, curves) to define control objective in an image space; and (ii) pose-based visual servoing [34], [35] which assumes the availability of the estimate of a target 6D pose. Some works propose switching between image- and pose-based servoing depending on the phase of movement [36]. Although image-based servoing allows to naturally incorporate visibility constraints in the control law, pose-based servoing is closer to applications such as object grasping [5]. We address the challenge of obtaining robust and fast estimates of 6D poses for pose-based control. These works typically implement control laws at the joint velocity level, which lack the natural impedance of torque-based control.

Model predictive control for visual servoing. For image-based visual servoing, MPC has shown superior performance to traditional reactive controllers [37], [38]. System dynamics in an image space is either approximated analytically [37], [38] or computed through learning-based optical flow estimation methods [39]. Image-based MPC servoing was successfully applied to control drone [40], legged platform [18], or mobile manipulator [41], [42]. Contrary to the state-of-the-art methods using image space MPC, we propose using MPC for pose-based control, where 6D pose is obtained by the proposed perception module.

III. OBJECT POSE GUIDED MODEL PREDICTIVE CONTROL

The objective of the proposed system is to perform manipulation tasks with respect to the 6D pose of a dynamically moving object using a camera mounted on a robot. The key technical challenge in such situations addressed by our method lies in achieving an accurate 6D pose estimation without introducing a significant delay into the control loop. The estimated 6D pose is then used in combination with MPC to achieve optimal control of the robot.

The proposed method is a 1 kHz torque level MPC controller taking reference from 6D object poses obtained from the 30 Hz image stream. To achieve this real-time robot control performance, the perception and control modules run asynchronously, as shown in Fig. 2. The perception module detects objects of interest in the scene and tracks them in a fast and temporally consistent manner as described in

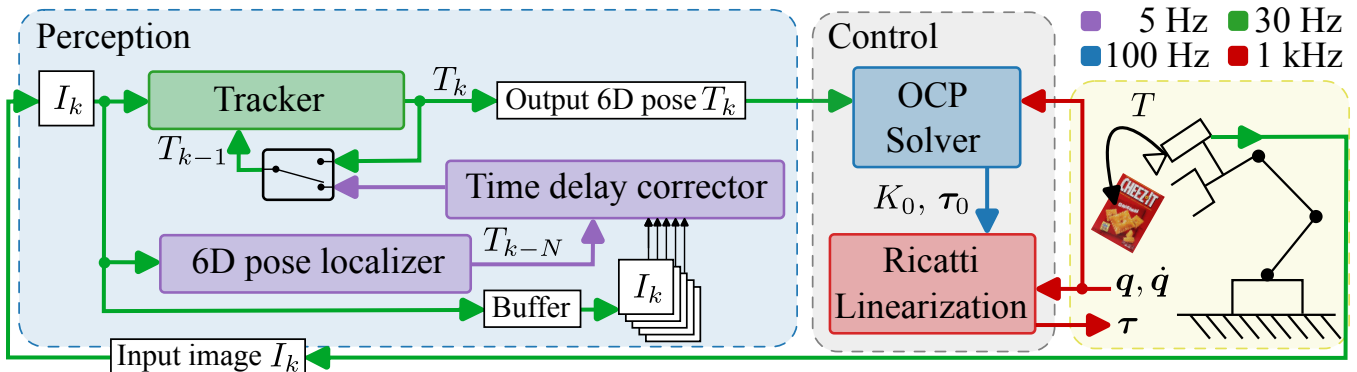


Fig. 2: **Overview of the perception-control cycle.** The objective of the feedback control is to track 6D pose of an object seen by a camera, as illustrated on the right by a robot and red cheez-it box. To achieve that, we designed a perception module that runs a fast local **Tracker** on an input image I_k with the initial pose T_{k-1} selected either from the previous run of the tracker or from the **6D pose localizer & Time delay corrector** modules, if that information is available. The **6D pose localizer** is slow and the objective of the **Time delay corrector** is to *catch-up* in time by quickly tracking through images stored in the buffer while the 6D pose localizer was computing. The output of the tracker, the pose T_k , is used by the **OCP solver** to compute Ricatti gains K_0 and torques τ_0 that are used by the **Ricatti Linearization** module to provide fast feedback for real-time robot control. Typical processing frequencies of individual modules are 5 Hz for the 6D pose localizer and the time delay corrector, 30 Hz for the camera and tracker, 100 Hz for the OCP solver, and 1 kHz for real-time robot control.

Sec. III-A. The key innovation is handling the inherent asynchronicity of the accurate-but-slow 6D pose localizer and fast-but-local tracker via the *time delay corrector* module that operates on the buffered images in order to *catch-up* in time. The 6D poses of objects detected by the perception module are used by the Optimal Control Problem (OCP) [43] solver and a feedback controller using a Ricatti Gains linearization of the solution [44] to compute torques for the robot at 1 kHz, as described in Sec. III-B. In the following sections, I symbols represent RGB image frame, while $T \in \mathbf{SE}(3)$ is a rigid body transformation.

A. Temporally consistent 6D object pose tracker

The objective of the perception module is to compute the 6D poses of objects in the scene based on the input image I_k , observed at discrete time k while introducing as little delay as possible. Although the proposed method can track an arbitrary number of objects, we describe the tracking of a single object pose T_k to simplify the notation.

6D pose localizer. With unlimited computational resources, the pose of an object can be estimated by the 6D pose localizer $T_k = f_{\text{localize}}(I_k)$ that detects the object of interest in the image and estimates its 6D pose with respect to the camera coordinate frame and, as a consequence, also the robot coordinate frame, as we assume the camera is calibrated with respect to the robot. However, robust object localizers are slow, *e.g.* 0.25 s for CosyPose [10] or even 30 s for MegaPose [13] on up-to-date hardware (see Sec. IV). The long computation time makes the localizer impractical for closed-loop control.

Tracker. To mitigate this limitation, we combine the localizer with a fast local tracker $T_k = f_{\text{track}}(I_k, T_{k-1})$ that computes the pose of objects from a given image I_k and initial guess of the pose T_{k-1} . Compared to the localizer, a single pass of

the tracker is fast, introducing only a few milliseconds delay into the system. However, it acts only locally, and it thus requires an initial pose that is refined based on the observed image. The tracker is not able to discover the presence of new objects, nor does it detect that the object is no longer visible by the camera when it is, for example, occluded.

Object localization and tracking (OLT). We combine the localizer and the tracker into a single perception module $T_k = f_{\text{OLT}}(I_k, T_{k-1})$ that computes fast feedback at the frequency of f_{track} while running f_{localize} in parallel for object (re-)discovery and more accurate pose estimation. Our architecture, shown in the perception plate of Fig. 2, runs $f_{\text{track}}(I_k, T_{\text{init}})$ on the current image with the initial pose T_{init} selected either from (i) the previous iteration of the tracker, *i.e.* T_{k-1} or (ii) the separate process that localizes the object if that information has already been computed by the *time delay corrector* for the previous image I_{k-1} , *i.e.* end of the image buffer. The main tracker is initialized once the first frame to enter the system has been processed by the localizer and time delay corrector.

Time delay corrector. In the parallel process, a single instance of the localizer is run all the time the resources are available. Let us assume that the localizer started processing input image I_{k-N} at time $k - N$. It takes some time to get output of f_{localize} during which new images arrive and are stacked inside a buffer. Once the pose $T_{k-N} = f_{\text{localize}}(I_{k-N})$ is computed, a second instance of the tracker is run on all images inside the buffer, *i.e.* $T_i = f_{\text{track}}(I_i, T_{i-1})$ iteratively for $i \in \{k - N + 1, \dots, k - 1\}$ while providing the final pose computed at the time $k - 1$ to the main tracker process. The timeline of the perception module is illustrated in Fig. 3. Note that our architecture assumes that the frequency of $f_{\text{track}}(\cdot)$ is higher than the frequency of the input image stream. Otherwise, the localizer process would

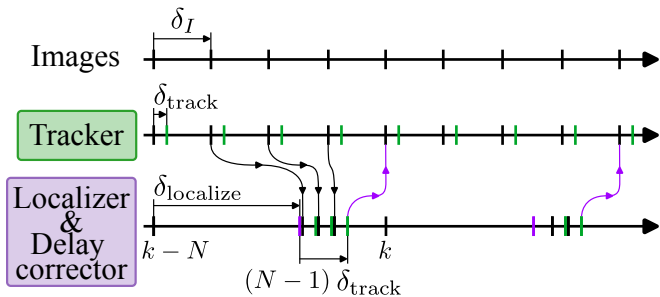


Fig. 3: **Perception module timeline.** The first row illustrates the stream of images with typical delay between images δ_I being 33 ms. The second row illustrates the delay caused by the **tracker** module (*i.e.* f_{track}), denoted by δ_{track} that corresponds to a few milliseconds and therefore output poses (**green ticks**) are produced at the frequency of the input image stream. The tracker needs initial pose that is taken either from previous run of the tracker or from the **6D pose localizer & time delay corrector** modules if possible as indicated by **purple arrows**. The 6D pose localizer runs f_{localize} (with typical δ_{localize} being a few hundreds of milliseconds) followed by f_{track} applied $N-1$ times on the buffered images (**green ticks in the third row**).

never *catch-up* with the main process. With the state-of-the-art tracker [14], this feedback can be easily calculated for image frequencies up to 120 Hz. However, the higher the input image frequency, the longer it takes to inject information from the localizer process. This affects the tracking accuracy as we analyze in Sec. IV.

B. 6D pose-based visual servoing using MPC

We design a controller that brings the camera attached to the robot end-effector to a user-defined relative pose w.r.t. the object pose T_k obtained from the object tracker. As a practical application, one may choose a reference pose from a set of predefined grasp poses for a given object. The challenge of the control lies in a real-time requirement of the robot to receive torque commands at 1 kHz. To address this challenge, we build on [44] and split the control into solving the optimal control problem at 100 Hz and computing 1 kHz feedback through Ricatti linearization, but here incorporating the 6D object poses as a guiding reference in the problem formulation.

OCP solver. The control module’s main objective is to follow the object given the latest object pose estimates T_k provided by the perception and the current robot state $\mathbf{x} = (\mathbf{q}^\top \quad \dot{\mathbf{q}}^\top)^\top$, with \mathbf{q} and $\dot{\mathbf{q}}$ being the measured joint angles and velocities, respectively. We control the manipulator at the torque level (*i.e.* $\mathbf{u} = \boldsymbol{\tau}$) to be able to exploit the natural dynamics of the manipulator and obtain smoother motions. Solving the Optimal Control Problem (OCP) produces optimal state and control trajectories over a fixed time horizon, where optimality is defined by a set of weighted high-level objectives. The resolution of the OCP is done in the framework of Differential Dynamic Programming (DDP) [45] and is implemented using the Feasibility-driven

DDP solver [43]. The OCP is transcribed to a nonlinear program by discretizing the continuous problem using a direct multiple-shooting strategy:

$$\begin{aligned} \arg \min_{\substack{\mathbf{u}_0, \dots, \mathbf{u}_{M-1} \\ \mathbf{x}_1, \dots, \mathbf{x}_M}} \sum_{i=0}^{M-1} l_i(\mathbf{x}_i, \mathbf{u}_i) + l_M(\mathbf{x}_M), \\ \text{s.t. } \mathbf{x}_{i+1} = f(\mathbf{x}_i, \mathbf{u}_i), \forall i \in \{0, \dots, M-1\}, \\ \mathbf{x}_0 = \hat{\mathbf{x}}, \end{aligned} \quad (1)$$

where $\hat{\mathbf{x}}$ is the latest robot state measurement, \mathbf{x}_i and \mathbf{u}_i are the state of the robot and the applied control at discrete time i , $f(\mathbf{x}_i, \mathbf{u}_i)$ describes the robot dynamics (*i.e.* articulated body algorithm), and l_i and l_M are running and terminal costs, respectively. The tracking objective of the control is specified by the costs, as we describe in Sec. III-C. Given an initial guess, the DDP algorithm solves (1) and returns a sequence of states and control actions by iterating Bellman recursions (see [43] for more details).

Ricatti linearization. Besides trivial systems and a short time horizon, it is impossible to solve OCP at the robot control frequency, *i.e.* 1 kHz. However, it has been shown [44] that the Ricatti gains K_0 obtained as a byproduct of the OCP solution can be used to implement a first-order approximation of the optimal policy. Denoting by $\boldsymbol{\tau}_0 = \mathbf{u}_0^*$ the first step of the optimal control obtained from the OCP solver, the linear approximation of the optimal policy is:

$$\boldsymbol{\tau}(\mathbf{x}) = \boldsymbol{\tau}_0 + K_0(\mathbf{x} - \mathbf{x}_0), \quad (2)$$

where \mathbf{x} is the latest robot state measurement and \mathbf{x}_0 is the state of the robot for which the OCP solution was found. This computation is immediate, it decouples the OCP problem complexity from the real-time constraints, and it allows us to solve long-horizon problems.

C. Tracking objective

The behavior of the controller is defined by the formulation of the running and terminal costs in eq (1). For our tracking problem, we formulate the costs as follows:

$$\begin{aligned} l_i(\mathbf{x}_i, \mathbf{u}_i) &= w_v l_v(\mathbf{x}_i) + l_x(\mathbf{x}_i) + l_u(\mathbf{x}_i, \mathbf{u}_i), \\ l_M(\mathbf{x}_M) &= w_v l_v(\mathbf{x}_M) + l_x(\mathbf{x}_M), \end{aligned} \quad (3)$$

where $l_v(\cdot)$ is the tracking cost scaled by w_v and $l_x(\cdot)$ and $l_u(\cdot)$ are state and control regularization costs, respectively.

Tracking cost. We define a tracking cost to minimize the **SE(3)** distance between the estimated pose of the object and the reference pose of the object T_{ref} , both expressed in the robot base frame, *i.e.*:

$$l_v(\mathbf{x}) = \left\| \log \left((T_{\text{BC}}(\mathbf{q}_k) T_k)^{-1} T_{\text{BC}}(\mathbf{q}) T_{\text{ref}} \right) \right\|^2, \quad (4)$$

where $T_k = f_{\text{OLT}}(I_k)$ is object pose estimated by the proposed tracker, $T_{\text{BC}}(\cdot)$ represents the forward kinematics from the robot base to the camera, and the operator \log represents the **SE(3)** log map [46]. The tracking cost approaches zero if the transformation between the robot camera and estimated object pose approaches T_{ref} .

State regularization cost. We define the cost of state regularization as $l_x(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_{\text{rest}})^\top Q_x (\mathbf{x} - \mathbf{x}_{\text{rest}})$ with $\mathbf{x}_{\text{rest}} = (\mathbf{q}_{\text{rest}}^\top \mathbf{0}^\top)^\top$ penalizing joint configurations far from a fixed rest configuration \mathbf{q}_{rest} and penalizing high joint velocities at the same. The objective of the regularization cost is to prevent robot *null-space* motion, *i.e.* motion that does not affect the pose of the camera itself. It is required for redundant robots, where the number of DoF for robot is higher than the task-space number of DoF. Regularization of the joint velocity prevents the solver from computing motions that are too aggressive. We set \mathbf{q}_{rest} to be the first configuration read after starting the controller.

Control regularization cost. The control regularization, achieved by $l_u(\mathbf{x}, \mathbf{u}) = (\mathbf{u} - \mathbf{u}_{\text{rest}}(\mathbf{x}))^\top Q_u (\mathbf{u} - \mathbf{u}_{\text{rest}}(\mathbf{x}))$, regularizes the controls so that they are not far from $\mathbf{u}_{\text{rest}}(\mathbf{x})$, where $\mathbf{u}_{\text{rest}}(\mathbf{x})$ is a torque that compensates for gravity at the robot configuration \mathbf{x} .

IV. EXPERIMENTS

In this section, we first quantitatively evaluate the proposed perception module on the YCBV dataset [20] that contains standardized objects that we also use in the second part of the section for the 6D pose-guided feedback control task on a real Franka Emika Panda robot. For the implementation of the localizer, we use CosyPose [10] and for the tracker we use ICG [14] unless specified otherwise.

A. Quantitative evaluation of the perception module

We quantitatively evaluate the new perception module on the YCBV dataset [20] using the 6D object pose (BOP benchmark) evaluation metrics [9]. The YCBV dataset consists of several videos of a moving camera showing a subset of 22 objects available in the dataset. Every frame of the video is annotated with the ground truth poses for all objects visible in the scene. We use the YCBV dataset because of the availability of the real objects for real-world experiments and of the pre-trained models for the CosyPose [10] object pose estimator. We use the BOP toolkit [9] to compute standard 6D pose error metrics to assess the quality of pose estimates. The evaluation procedure feeds the images of the input video sequence in order and with a given frequency to the perception module. The output poses are compared with the ground truth by evaluating *BOP Average Recall* score defined in [9]. The results of the evaluation procedure are shown in Fig. 4 and discussed next.

Evaluation baselines. There are two main baselines shown in the plot: (i) *Localizer*, that runs localizer on every frame of the video, and (ii) *Tracker-InitLocalizer* that runs the tracker with initialization computed by the localizer on the first frame of the video. Both baselines are shown as horizontal lines as they were evaluated independently on the frequency of the image stream. The *Localizer* is introducing high time delay in the system and, therefore, is not suitable for closed-loop control. However, the results show that the localizer is accurate. The *Tracker*, on the other hand, runs online with only a small delay, but is not capable of (re-)discovering new or lost object tracks. Therefore, its average recall is small.

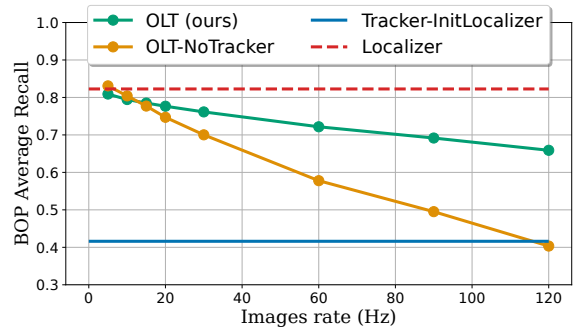


Fig. 4: Average recall (higher is better) of BOP metrics [9] measuring the accuracy of 6D pose estimation of different implementations of object localization and tracking. The comparison was run on the YCBV video dataset replayed at different frequencies on the same hardware.

OLT evaluation. Our method (*OLT*), lies in between the two baselines. It runs online with the same delay as *Tracker* but also achieves the *Localizer*'s recall for the low frequencies of the input image stream. The average recall drops with increasing frequency as the output pose of the localizer is injected into the tracker less frequently. Asymptotically, for image stream frequency approaching the tracker computation frequency, the *OLT*'s recall would approach performance of pure tracker as the time delay corrector would never *catch-up* in time with the tracker process.

OLT without tracker. To assess the influence of the tracker, we perform an ablation study in which we redefine the tracker as identity mapping, *i.e.* $T_k = f_{\text{track}}(I_k, T_{k_1}) := T_{k-1}$. The recall obtained for various image stream frequencies is shown as *OLT-NoTracker* curve in Fig. 4. It can be seen that the influence of the ICG tracker (*i.e.* *OLT* (ours)) increases with frequency compared to the identity tracker. Therefore, the local tracker plays an important role during the computing time of the localizer. Note that this effect is more pronounced for fast-moving objects which are not present in the YCBV dataset.

Replicability of the results. As shown in Fig. 4, the average recall of our method depends on the input image

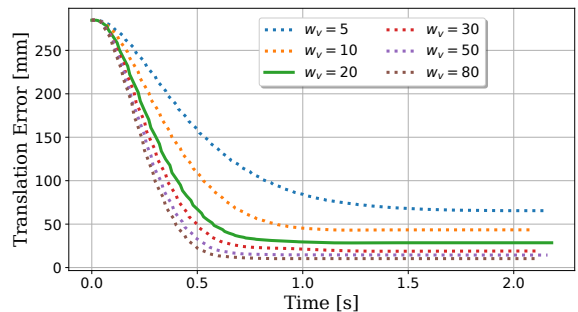


Fig. 5: Step-response of the MPC (without the perception) after simulating the target pose rotation by 30 degrees.



Fig. 6: **Visualization of the step-response experiment.** The first row shows the images captured by the camera mounted on the robot together with the projection of the pose estimated by our perception module (green contour) and the projection of the target reference pose (blue contour). The second row shows the images captured by an external camera depicting the motion of the robot. The goal of the controller is to move the robot end-effector to a given fixed relative pose w.r.t. the detected object pose. The initial configuration of the manipulator is intentionally set away from the target to evaluate the step response of the system. The controller brings the camera close to the desired reference pose in 1.5 seconds. **Please see the supplementary video for additional examples and experimental analysis.**

stream frequency, as the localizer produces a more accurate 6D pose less often. Therefore, the computed values also depend on the hardware, and thus are only comparable when run on comparable setups. We evaluated all methods on the same computer equipped with 12 cores *AMD® Ryzen Threadripper PRO 3945WX* CPU and a *NVIDIA GeForce RTX 3080* GPU.

B. Visually guided feedback control

We experimentally validate the design of our 6D object perception module together with the MPC controller using the following experimental setup. We use a 7 DoF Franka Emika Panda robot equipped with a RealSense D455 camera attached to its end-effector (eye-in-hand configuration). The camera mounting w.r.t. end-effector was calibrated. We configured the camera to produce an RGB video stream at 30 Hz with a 640x480 resolution. We control the Panda robot in torque-level control mode that requires commands to be sent at 1 kHz frequency. This justifies the use of the Ricatti Linearization module, which guarantees that a torque command will be computed at this rate. The OCP is solved with Crocoddyl [43] which uses the efficient robot dynamics implementation from Pinocchio [47]. To model the dynamics of the robot, we use inertial parameters from [48]. Computations of the perception module and OCP solving is handled by a computer described in Sec. IV-A. The 1 kHz Ricatti linearization control loop is computed on another real-time-preempted computer to guarantee the response time requirements of the Panda robot. The communication between the real-time and the non-real-time computer is implemented using the robotic operating system (ROS) [49].

MPC control evaluation. To assess the quality of the MPC control, we perform an experiment on a Panda robot where we analyze the step response of the controller after artificially rotating the target 6D pose by 30 degrees, *i.e.* the perception

module is not used in this experiment. The evolution of the translation tracking error is shown in Fig. 5 for different values of tracking weight w_v (see eq. (3)). The results confirm that the tracker converges towards the target with steady-state error depending on the tracking weight. The orientation error (not shown) follows the same pattern. Based on the results, we have chosen the tracking weight w_v to be equal to 20 since the steady-state error is acceptable for our task and a lower weight leads to less aggressive behavior of the controller. The other weights in the cost were set as $Q_x = \text{diag}(0.3, \dots, 0.3, 3, \dots, 3)$ and $Q_u = \text{diag}(0.1, \dots, 0.1)$. **MPC tracking validation.** We set up a closed-loop robot control experiment shown in Fig. 6 in which the perception and control modules enable us to bring the end-effector of the robot to a desired reference pose with respect to a YCBV object. Despite the relatively fast motion of the end effector and the presence of specular reflections on the object surface, the tracker is able to maintain an accurate estimation of the object pose throughout the trajectory. More examples are presented in the accompanying video.

V. CONCLUSION

Accurate and low-latency object pose estimation is necessary to enable robot interaction with dynamically moving objects, for example, in human-robot handover tasks. Our work shows that a high-accuracy but slow 6D pose localizer and fast frame-to-frame 6D pose object trackers can be combined to obtain low latency (< 5 ms) pose estimates. The proposed algorithm has been validated through both (i) a quantitative study on a benchmark of common household objects and (ii) by developing an MPC-based object pose tracking feedback controller. This work opens up the possibility of visually guided manipulation in 3D dynamic environments, for example, in human-robot collaboration or mobile robot manipulation, without the need for fiducial markers or motion capture systems.

REFERENCES

- [1] P. Furgale and T. D. Barfoot, "Visual teach and repeat for long-range rover autonomy," *Journal of field robotics*, vol. 27, no. 5, pp. 534–560, 2010.
- [2] F. Chaumette, S. Hutchinson, and P. Corke, "Visual servoing," *Springer handbook of robotics*, pp. 841–866, 2016.
- [3] T. Chabal, R. Strudel, E. Arlaud, J. Ponce, and C. Schmid, "Assembly planning from observations under physical constraints," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10223–10229, IEEE, 2022.
- [4] V. Ortenzi, A. Cosgun, T. Pardi, W. P. Chan, E. Croft, and D. Kulić, "Object handovers: a review for robotics," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1855–1873, 2021.
- [5] S. S. Mirrazavi Salehian, N. Figueroa, and A. Billard, "A unified framework for coordinated multi-arm motion planning," *The International Journal of Robotics Research*, vol. 37, no. 10, pp. 1205–1232, 2018.
- [6] N. Figueroa, S. Faraji, M. Koptev, and A. Billard, "A dynamical system approach for adaptive grasping, navigation and co-manipulation with humanoid robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7676–7682, 2020.
- [7] J. Haviland, N. Sünderhauf, and P. Corke, "A holistic approach to reactive mobile manipulation," *IEEE Robotics and Automation Letters*, vol. 7, p. 3122–3129, Apr 2022.
- [8] T. Hodaň, M. Sundermeyer, B. Drost, Y. Labbé, E. Brachmann, F. Michel, C. Rother, and J. Matas, "Bop challenge 2020 on 6d object localization," in *Computer Vision—ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pp. 577–594, Springer, 2020.
- [9] M. Sundermeyer, T. Hodaň, Y. Labbe, G. Wang, E. Brachmann, B. Drost, C. Rother, and J. Matas, "Bop challenge 2022 on detection, segmentation and pose estimation of specific rigid objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2784–2793, 2023.
- [10] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "Cosypose: Consistent multi-view multi-object 6d pose estimation," in *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*, pp. 574–591, Springer, 2020.
- [11] R. L. Haugaard and A. G. Buch, "Surfemb: Dense and continuous correspondence distributions for object pose estimation with learnt surface embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6749–6758, 2022.
- [12] L. Lipson, Z. Teed, A. Goyal, and J. Deng, "Coupled iterative refinement for 6d multi-object pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6728–6737, 2022.
- [13] Y. Labbé, L. Manuelli, A. Mousavian, S. Tyree, S. Birchfield, J. Tremblay, J. Carpentier, M. Aubry, D. Fox, and J. Sivic, "Megapose: 6d pose estimation of novel objects via render & compare," *arXiv preprint arXiv:2212.06870*, 2022.
- [14] M. Stoiber, M. Sundermeyer, and R. Triebel, "Iterative corresponding geometry: Fusing region and depth for highly efficient 3d tracking of textureless objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6855–6865, 2022.
- [15] M. Fiala, "Artag, a fiducial marker system using digital techniques," in *CVPR*, 2005.
- [16] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, 2014.
- [17] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *ICRA*, 2011.
- [18] R. Parosi, M. Risiglione, D. G. Caldwell, C. Semini, and V. Barasuol, "Kinematically-decoupled impedance control for fast object visual servoing and grasping on quadruped manipulators," *arXiv preprint arXiv:2307.04918*, 2023.
- [19] E. Dantec, M. Naveau, P. Fernbach, N. Villa, G. Saurel, O. Stasse, M. Taix, and N. Mansard, "Whole-body model predictive control for biped locomotion on a torque-controlled humanoid robot," in *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, pp. 638–644, IEEE, 2022.
- [20] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols," *arXiv preprint arXiv:1502.03143*, 2015.
- [21] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.
- [22] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, "Deepim: Deep iterative matching for 6d pose estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 683–698, 2018.
- [23] C. Harris and C. Stennett, "Rapid-a video rate object tracker.," in *BMVC*, pp. 1–6, 1990.
- [24] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 2, pp. 1508–1515, Ieee, 2005.
- [25] A. I. Comport, E. Marchand, M. Pressigout, and F. Chaumette, "Real-time markerless tracking for augmented reality: the virtual visual servoing framework," *IEEE Transactions on visualization and computer graphics*, vol. 12, no. 4, pp. 615–628, 2006.
- [26] S. Trinh, F. Spindler, E. Marchand, and F. Chaumette, "A modular framework for model-based visual tracking using edge, texture and depth features. in 2018 IEEE," in *RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 89–96.
- [27] B. Rosenhahn, T. Brox, and J. Weickert, "Three-dimensional shape knowledge for joint image segmentation and pose tracking," *International Journal of Computer Vision*, vol. 73, pp. 243–262, 2007.
- [28] V. A. Prisacariu and I. D. Reid, "Pwp3d: Real-time segmentation and tracking of 3d objects," *International journal of computer vision*, vol. 98, pp. 335–354, 2012.
- [29] V. A. Prisacariu, O. Kähler, D. W. Murray, and I. D. Reid, "Real-time 3d tracking and reconstruction on mobile phones," *IEEE transactions on visualization and computer graphics*, vol. 21, no. 5, pp. 557–570, 2014.
- [30] W. Kehl, F. Tombari, S. Ilic, and N. Navab, "Real-time 3d model tracking in color and depth on a single cpu core," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 745–753, 2017.
- [31] M. Stoiber, M. Pfanne, K. H. Strobl, R. Triebel, and A. Albu-Schäffer, "Srt3d: A sparse region-based 3d object tracking approach for the real world," *International Journal of Computer Vision*, vol. 130, no. 4, pp. 1008–1030, 2022.
- [32] L. Weiss, A. Sanderson, and C. Neuman, "Dynamic sensor-based control of robots with visual feedback," *IEEE Journal on Robotics and Automation*, vol. 3, no. 5, pp. 404–417, 1987.
- [33] J. T. Feddema and O. R. Mitchell, "Vision-guided servoing with feature-based trajectory generation (for robots)," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 5, pp. 691–700, 1989.
- [34] W. J. Wilson, C. W. Hulls, and G. S. Bell, "Relative end-effector control using cartesian position based visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 684–696, 1996.
- [35] B. Thuilot, P. Martinet, L. Cordesses, and J. Gallice, "Position based visual servoing: keeping the object in the field of vision," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 2, pp. 1624–1629, IEEE, 2002.
- [36] J. Haviland, F. Dayoub, and P. Corke, "Control of the final-phase of closed-loop visual grasping using image-based visual servoing," *arXiv preprint arXiv:2001.05650*, 2020.
- [37] M. Sauvée, P. Poignet, E. Dombre, and E. Courtial, "Image based visual servoing through nonlinear model predictive control," in *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 1776–1781, IEEE, 2006.
- [38] G. Allibert and E. Courtial, "What can prediction bring to image-based visual servoing?," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5210–5215, IEEE, 2009.
- [39] P. Katara, Y. Harish, H. Pandya, A. Gupta, A. Sanchawala, G. Kumar, B. Bhowmick, and M. Krishna, "Deepmpcvs: Deep model predictive control for visual servoing," in *Conference on Robot Learning*, pp. 2006–2015, PMLR, 2021.
- [40] M. Jacquet and A. Franchi, "Motor and perception constrained nmpc for torque-controlled generic aerial vehicles," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 518–525, 2020.
- [41] H. Bildstein, A. Durand-Petiteville, and V. Cadenat, "Visual predictive control strategy for mobile manipulators," in *2022 European Control Conference (ECC)*, pp. 1672–1677, IEEE, 2022.
- [42] H. Bildstein, A. Durand-Petiteville, and V. Cadenat, "Multi-camera visual predictive control strategy for mobile manipulators," in *2023 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 476–482, IEEE, 2023.

- [43] C. Mastalli, R. Budhiraja, W. Merkt, G. Saurel, B. Hammoud, M. Naveau, J. Carpentier, L. Righetti, S. Vijayakumar, and N. Mansard, "Crocodyl: An efficient and versatile framework for multi-contact optimal control," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2536–2542, IEEE, 2020.
- [44] E. Dantec, M. Taix, and N. Mansard, "First order approximation of model predictive control solutions for high frequency feedback," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4448–4455, 2022.
- [45] D. Mayne, "A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems," *International Journal of Control*, vol. 3, no. 1, pp. 85–95, 1966.
- [46] J. Sola, J. Deray, and D. Atchuthan, "A micro lie theory for state estimation in robotics," *arXiv preprint arXiv:1812.01537*, 2018.
- [47] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiroux, O. Stasse, and N. Mansard, "The pinocchio c++ library – a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *IEEE International Symposium on System Integrations (SII)*, 2019.
- [48] C. Gaz, M. Cognetti, A. Oliva, P. R. Giordano, and A. De Luca, "Dynamic identification of the franka emika panda robot with retrieval of feasible parameters using penalty-based optimization," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4147–4154, 2019.
- [49] M. Quigley, "Ros: an open-source robot operating system," in *IEEE International Conference on Robotics and Automation*, 2009.