IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan. Cite as T-ASE paper.

Humanoid Loco-Manipulations using Combined Fast Dense 3D Tracking and SLAM with Wide-Angle Depth-Images

Kevin Chappellet, Masaki Murooka, Guillaume Caron, *Senior Member, IEEE,* Fumio Kanehiro, *Member, IEEE,* and Abderrahmane Kheddar, *Fellow, IEEE*

***Abstract*—To efficiently achieve complex humanoid loco-manipulation tasks in industrial contexts, we propose a combined vision-based tracker-localization interplay integrated as part of a task-space whole-body optimization control. To achieve good perception complementarity between manipulation and localization, a new fast dense 3D model-based tracking using wide-angle depth image is developed and used in conjunction with a simultaneous localization and mapping software. Our approach allows humanoid robots, targeted for industrial manufacturing, to manipulate and assemble large-scale objects while walking. It is assessed with experiments consisting in rolling and assembling in an unwinder a heavy and wide bobbin using bimanual grasping and bipedal locomotion at a time. This experimental use-case is found in some large-scale manufacturing where bobbins are enrolled with various materials (cables, papers, rubbers, etc.). The same experiments are made using two different humanoid robots of the same family.**

***Index Terms*—Loco-manipulation, Manufacturing humanoids, vSLAM, 3D object tracking.**

NOTE TO PRACTITIONERS

This paper aims at deploying humanoid robots in large-scale manufacturing industries. We consider non-added value tasks related to transporting large tools or objects such as large bobbins by means of locomanipulation skills, similarly to human workers. We developed a task-space control framework that has been successfully applied in the aircraft industry. In the frame of a current collaboration with other major industrial sectors, we enhanced our control framework to interplay between SLAM and visual tracking to realize robust loco-manipulation tasks. Our approach can be applied and ported to any humanoid robot or bi-manual wheeled mobile robots with minor programming effort as the software is made open. Preliminary experiments with two different humanoids and use-cases suggest that our approach is feasible. In future research, we will address the problem of performance to reach at least human-speed in the execution of locomanipulation tasks in large-scale industry and automation contexts.

Manuscript received 05 Dec. 2022; revised 13 Apr. 2023; accepted 27 May 2023.

The authors are with the CNRS-AIST JRL (Joint Robotics Laboratory), IRL, National Institute of Advanced Industrial Science and Technology (AIST), Japan.

K. Chappellet is also with University of Montpellier, Montpellier, France.

A. Kheddar is also with the CNRS-University of Montpellier, LIRMM, Montpellier, France.

G. Caron is also with Université de Picardie Jules Verne, MIS lab, France.

This paper has supplementary video.

Digital Object Identifier 10.1109/TASE.2023.3283497

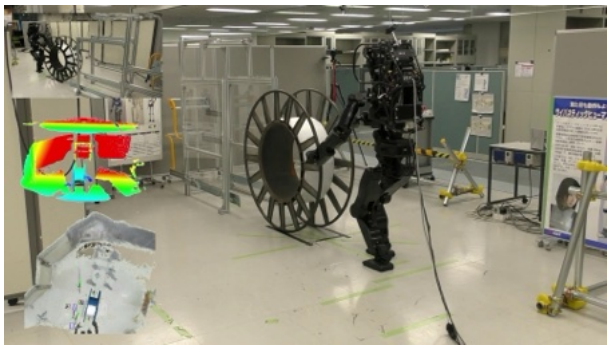


Fig. 1: HRP-5P loco-manipulation of a bobbin relying on vision to move toward an unwinder. Left, top to bottom: external view, tracking visualization, vSLAM top view.

I. INTRODUCTION

OUR recent results in humanoid automation for aircraft manufacturing [1] revealed that in large-scale industrial settings, reliably localizing both the robot within the shop-floor and the manipulated object w.r.t the robot is important to achieve closed-loop localization and visual servoing assemblies. This is particularly challenging when the robot moves together with an object that it manipulates [2].

Nowadays, visual SLAM (Simultaneous Localization and Mapping) proves to be a well-documented technology for robot localization at large, e.g., [3], [4]. SLAM has very known shortcomings when the environment features moving objects, or is poorly textured, or contains confined spaces, or when an object obstructs permanently the acquired video stream [5]. As a first example leveraging the state-of-the-art, consider a humanoid robot driving a car [6]. The car appears in the field of view of the camera mounted on the head of the humanoid. In fact, it can even take a large portion of the image. In this case, the car is an outlier for the SLAM, which is operating to localize the car in the surrounding area, assuming the robot can localize itself inside the car. A similar example, that we tackle in this article, is a humanoid robot moving a large object through manipulation while walking (loco-manipulation). Here, the manipulated object is an outlier for the SLAM that localizes the robot in its environment. In brief, the images obtained from the robot's embedded camera contains both the surrounding area and part of some *outliers of interest*, that are respectively the car and the manipulated object in the previous examples.

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.

Ideally, the SLAM would automatically extract features from the image corresponding only to the environment map. When the outliers of interest are static w.r.t. the robot camera (i.e., they do not move in the image over time), a mask can be applied as in [7] to ignore them from the image. Moreover, such a mask is easy to design in some non-static cases too, such as when the image outliers are limbs of the robot [8]. Hence, when the outliers of interest move w.r.t. the camera frame, e.g., because the robot is not attached to the car or the manipulated objects are not attached to the robot, they need to be efficiently recognized and tracked to be ignored during the SLAM process [9].

In the previous examples we provide, it is important to track the car w.r.t. the camera frame for the robot to manipulate the car accessories to drive it. Similarly, it is also important for the robot to know where it stands w.r.t. the object to be manipulated or *vice-versa*. In this case, current SLAM technology cannot be used to localize both the object and the robot. We also encountered in [1] situations where the humanoid evolves in a narrow and confined space, where SLAM can fail because of the lack of textures.

The challenge that has not been considered in humanoid loco-manipulation, is to allow humanoid robots to use the image acquired by its embedded camera for both locating itself w.r.t. its surrounding and locating the manipulated objects or systems that move w.r.t. the robot. Moreover, since such objects are close to the sight of the robot and within its reach, they can take a large part of the image compared to the surrounding environment. In this case, wide-angle cameras are the best sensors for both visual SLAM and visual tracking.

Numerous camera-vision-based tracking and localization algorithms work quite well using various classical cameras such as Asus Xtion Pro Live or Microsoft Kinect (see a benchmark study in [7]). With the recent release of the Microsoft Azure Kinect camera, existing visual tracking, and associated datasets became obsolete as they do not apply straightforwardly; it has a new kind of Time-of-Flight (ToF) depth sensor measuring depths in a wide-angle [10]. To our best knowledge, other existing wide-angle depth images are estimated from stereo fisheye cameras [11], e.g., IntelRealSense T265 [12].

We propose a new fast dense tracker for large objects pose estimation while manipulated by the robot that works with a wide-angle depth camera. This tracker is used concurrently with robot localization based on RTAB-Map [13] SLAM applied to the color-depth stream of the camera. Our approach allows localizing the robot in its environment and, at the same time, tracking an object (or several) while being manipulated, both during locomotion. If a trackable object is not moving and its position known w.r.t. the environment, the dense tracker can also be used to localize the robot [14]. This allows extending the closed-loop task-space `mc_rtc` controller [15], [16] to whole-body loco-manipulation tasks. We have performed experiments with two large objects, the hardest being inspired by a real industrial use-case we are investigating, and assess our approach with two different humanoid robots.

II. Related Work

We provide a brief overview of a few existing works related to the main components we are using: whole-body loco-manipulation, vSLAM and visual tracking.

A. Whole-body loco-manipulation

Loco-manipulation is a cross-topic between locomotion and manipulation where a robot has to interact and move its base at a time [17]. Loco-manipulation is challenging in terms of balance of both the robot base and the object [2]. For example, [18] uses a mobile base robot PR-2 to manipulate a cart in an indoor environment; it has to follow a predefined path in a 2D map while slightly turning the cart in narrow passageways. A single grasping sequence allows manipulating the cart. In [19], the humanoid robot HRP-2 is operating a wheelchair which pose is obtained from visual tracking. The robot follows instructions such as "Go Forward/Backward", and "Turn Right/Left" provided by the wheelchair user. In [20], simulations of NAO exhibit human-like motion to grasp an object by starting the grasping sequence while finishing to walk toward that object. This is obtained by extending a locomotion framework to a tasks overlapping framework. In [21], the authors proposed a method to generate a plan of locomotion and manipulation primitives. In [22], a whole-body kinematics approach to manipulate and move carts and wheelbarrows in diverse scenarios with a humanoid robot is presented; it is focused on forward motion. In [23], the humanoid HRP-5P realizes a plasterboard installation in an autonomous way; it relies on perception and accomplishes different motions such as grasping the plasterboard, locomotion toward the installation place and plasterboard's manipulation to fix it to a wall. In [24] a new planning loco-manipulation multi-contact tasks is proposed based on graph search and reachability maps; it is followed in [25] with a stabilization strategy for humanoid robots to perform under sustained forces applied on the object to be manipulated while walking. The loco-manipulation assumes the object is initially already grasped.

In this article, we leverage [25] for the whole-body loco-manipulation, enlarging its scope to situations where the humanoid does not already hold the object to manipulate from the start. To deal with such situations, works such as [22], [19], [21] have shown that visual pose estimation of the object to be manipulated can be considered to provide momentary feedback for object grasping. But beyond object pose estimation and tracking in the robot's egocentric frame, the robot needs to localize the object within its environment to accurately reach a target location for the manipulated object.

B. Visual SLAM

Many SLAM variants have been proposed exploiting various sensors as lidars, sonars, inertial measurement units (IMU), or cameras [26]. The use of a camera as a single (or main) sensor is called visual SLAM (vSLAM in short) [27]. The standard pipeline of vSLAM consists of three main components [28]: (i) camera pose tracking, (ii) scene mapping, and (iii) loop closing. Once the scene is mapped, the map can be reused for localization-only, see e.g., [1]. vSLAM systems have been made for various types of cameras, ranging from the most conventional in seminal vSLAM works [29], [30], to stereo vision [31], panoramic [32], 360 [33] and active RGB-D [34], [35], [36], [37] vision. It was recently shown that

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.

the use of an RGB-D camera leads to highest precision for both mapping at scale and localization stages when the visual information is made of sparse feature points [7]. The precision of camera pose tracking is even increased when considering dense direct information, i.e., pixel brightness [38]. It is at the price of a tighter basin of convergence. This is not a problem if the camera velocity is bounded w.r.t its acquisition frame-rate, the RGB and Depth streams are synchronized, and the camera calibration is accurate [28], [39].

We consider the dense and direct RGB-D vSLAM known as Real-Time Appearance-Based Mapping (or RTAB-Map in short) [13] because it is reliable and convenient as it has largely been evaluated by the research community and it is available open-source[1]. Indeed, RTAB-Map has recently been used with humanoid robots such as Pepper [40] and HRP-2Kai [41].

C. 3D visual tracking

It is challenging to make a fair coverage of the rich literature in robotic visual tracking. The works are diverse and some are optimized for different types of the many kinds of cameras that are nowadays available. We report however few recent works that are close to our circle of interest.

1) With a wide-angle camera: In [42], [43] a 3D model-based approach is proposed to track an object, relying on geometrical features such as lines, within panoramic images. Thanks to their wide field-of-view (FoV), panoramic cameras capture large parts of big objects (even buildings) within a single image. But these geometrical approaches, known to be computationally frugal, tend to lose track in case of fast motion due to motion blur. With an omnidirectional camera, a mobile robot is localized within roads modeled as a color point cloud by using image intensities [44]. While this method is accurate, it requires computing a 3D rendering at each step of the optimization making it computationally expensive, and it is tailored for textured objects or environments.

As we want our robots to concurrently manipulate or localize themselves to any objects, even textureless, these methods do not fulfill our requirements.

2) With an RGB-D camera: The use of RGB-D cameras is nowadays banal, the depth has the advantage of sensing many details on textureless objects w.r.t to their material, allowing efficient 3D model-based tracking [45], [46], [47]. Furthermore, the captured geometrical information close to the geometry of an object 3D model makes such a camera easy to use within robot manipulation applications [48], [49].

In this article, we use an RGB-D camera with a wide-angle depth frame to combine the best of both types. Although the previous off-the-shelf algorithms cannot directly be used, they have inspired our work significantly. Indeed, they all rely on depth frames captured with no distortions or so little compared to an almost hemispherical field-of-view that frames can be undistorted easily without impacting the resolution and thus run the same algorithms. The wide-angle depth capture needed in this work implies to track the 3D object in the captured frame featuring strong distortions for an accurate pose estimate in real-time, hence to adapt the tracking algorithm.

[1]The vSLAM system we used in [1] has been acquired by Apple and is not open anymore.

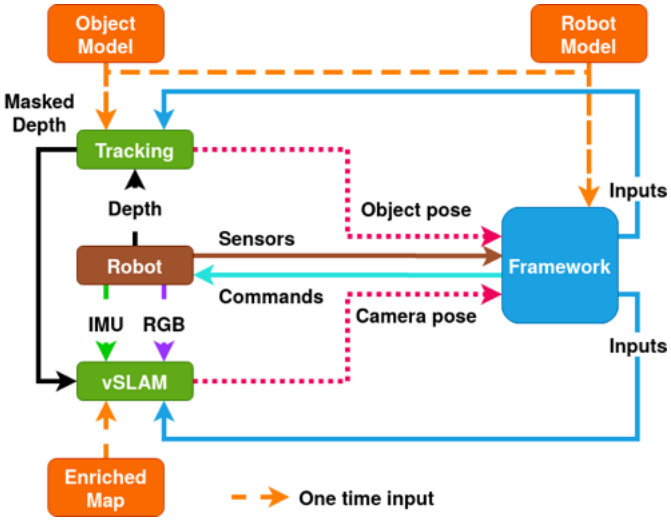


Fig. 2: Vision-based loco-manipulation architecture. Block colors are: brown for the hardware, blue for the control framework, green for vision, orange for data. The tracking module takes as input the manipulated objects and robot models once and camera depth data continuously; it outputs the objects of interest poses for the control framework (if the object is static in the scene it can be used to localize the robot). The tracking outputs also a mask to cancel moving parts of the image keeping only the static surrounding parts to input the vSLAM. The latter is initialized by the enriched map obtained off-line and also takes as input the robot IMU and camera RGB data continuously; it outputs the camera pose to localize the robot. The control framework manages all inputs and outputs for successful operation realization. It takes mainly robot states and sensors, poses and generates commands to the robot and manages the tracking and vSLAM.

III. Architecture of Vision-Based Loco-Manipulation

We introduce the architecture (Fig. 2), and its four components, used to achieve autonomous loco-manipulation of large-scale objects with a humanoid robot: (i) a humanoid robot (we experimented two different ones), (ii) our task-space optimization control framework, (iii) a vSLAM, and (iv) our object tracking that we detail later in Section IV. In addition, these components rely on three inputs in terms of data: (i) a robot model (URDF file), (ii) an object model (3D CAD model), and (iii) an enriched map explained in Section III-C.

A. Humanoid robot

The requirements for the humanoid robot are mainly twofold: manipulation capabilities, in addition to bipedal locomotion, and embedded vision sensor. Manipulation of large-scale objects requires: (i) grasping tools suitable to the tasks in an industrial context, (ii) force sensor at each end-effector, and (iii) an embedded IMU. To implement the manipulation of large-scale objects, our framework leverages [25] that achieve dynamic balance under sustained or varying external forces applied on the manipulated object (see Sec. III-B). For the visual perception (vSLAM and tracking), the robot relies on

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.

a unique camera. Because of the constraints inherent to the handling of large objects that do not have a rich texture and because of their closeness to the robot, an RGB-D camera with a wide-angle depth sensor is the best solution. The wide-angle depth-image is used by object tracking while the RGB and rectified depth images are used by vSLAM.

B. Task-space optimization control framework

Our controller is based on the open-source `mc_rtc`[2] open-source framework [50], [15]. It is a task-space optimization controller formulated as a weighted quadratic program (QP) with a full user-friendly development environment, i.e., many examples and tutorials for complex controllers use-cases with bridges and interfaces to main existing robotic simulators, QP solvers, logging tools, robotic models computations, etc.

At the level of task planning, current and next humanoid robot's *actions* are decided during the manipulation of an object in order for it to reach pre-defined waypoints (intermediate goals) in the environment. The map prebuilt with vSLAM contains the objects' targets (e.g., automaton systems or deposit spots, etc.). This map can be converted to a point-cloud that can be used for 3D registration of the targets. At the end of the registration, we have the targets' poses within the map. Our experience in aircraft manufacturing [1] highlighted execution robustness when the control is made directly in the vSLAM map space. Waypoints are defined only one time per environment, and are dynamically re-computed when the prebuilt map is used. The use of waypoints reflects in fact rules and practices defined in industries that inform where to go through or to be and how. Of course a full planning without waypoints, or automatically generated waypoints are both possible. But to obey safety constraints and practices, it is highly recommended to leave the definition of waypoints to the operator's responsibility.

In practice no new waypoints are generated during an experiment. By design, the robot should not move far away from the waypoints. The next action is computed as to reduce the error between the object and the current targeted waypoint. If the rotation part of the error is greater than some pre-defined tolerance (industry defined), the next action is to reduce this angle; i.e., to re-align the object with the waypoint. If the translation part of the error is greater than some pre-defined threshold, the next action is to reduce the distance; i.e., to move forward (or backward) the object.

In most of the use-cases an action is pre-defined among a set of the following task templates (with explicit labels): `Walk`, `WalkRelative`, `GraspObject`, `ReleaseObject` and `MoveObject`. All these actions rely on outputs from both vSLAM and object tracking components.

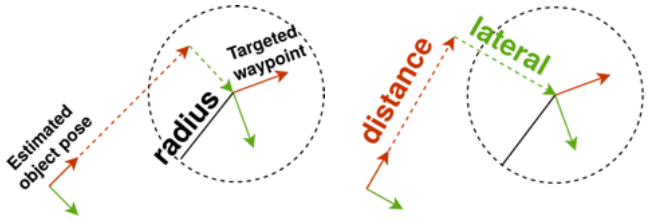
The `Walk` action allows the robot moving with free grippers. It relies on vSLAM or visual tracking for estimating robot's state to dynamically re-compute next footsteps toward the desired destination using predefined or computed path.

The `WalkRelative` action aligns the robot w.r.t. the object either while or without manipulating it. This alignment depends on the triplet *robot, object, next action to perform* and can be manually pre-defined or planned. As the object is estimated w.r.t. the robot, a footsteps plan is computed on the fly from the current robot pose to a pose on the object forward axis when this action is needed.

`GraspObject` and `ReleaseObject` use the estimated object pose, the object model, and pre-defined grasping poses expressed in the object model coordinate system. A target pose is dynamically computed to either grasp or release the object with the robot end-effector by transforming the pre-defined grasping poses using the current estimated pose. For example, considering as object type a bobbin (see Fig. 1), i.e., a rolling object with spokes similar to a bike wheel, the pre-defined grasping poses are set in-between consecutive pairs of spokes using the following bobbin properties: width, number of spokes and rolling radius. This allows to grasp the closest empty space between two spokes.

`MoveObject` is the core loco-manipulation behavior, it includes: `StraightObject`, `TurnObjectFacing` and `TurnObjectAlign` actions. Figure 3 illustrates the way the action is chosen w.r.t. to visual estimation between `StraightObject` and `TurnObjectFacing`. If the manipulated object is facing the waypoint, up to an angular tolerance Δ_θ (Fig. 3(a)), the robot moves it straight until falling below the distance threshold Δ_d; otherwise, (Fig. 3(b)), the robot turns the object to face the waypoint. `TurnObjectAlign` action is used to align the object with the waypoint orientation once the object reached the waypoint.

Our framework is open to any programming mode, at the choice of the operator. A mission can be programmed to be fully autonomous (i.e., full execution without intervention of the operator), or partially autonomous (i.e., adding phases where the operator intervene to program actions on the fly), or interactive (i.e., the operator programs tasks sequentially and validates the next task to be performed at will). For the bobbin experiments described in Sec. VI-C and VI-D, once the robot starts to move there are no new inputs from an operator. All is made by the control framework. The sequencing of tasks (i.e., adding or removing tasks in the controller cost function and constraints) is made on the basis of a programmed FSM (Finite State Machine) that monitors the task progression and related sensor state (those of the robot and any others). In the



(a) Object facing the waypoint: `StraightObject` is to go straight.

(b) Object not facing the waypoint: `TurnObjectFacing` is to turn on the spot.

Fig. 3: The `MoveObject` resulting action depends on the facing angle error, i.e., the angle between the current object main axis (red arrow) and the waypoint position, and on the distance between the current object position and the waypoint. "radius": threshold distance Δ_d; "lateral": tangential representation of the facing angle error Δ_θ.

[2] `mc_rtc` open-source framework: https://jrl-umi3218.github.io/mc_rtc.

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.

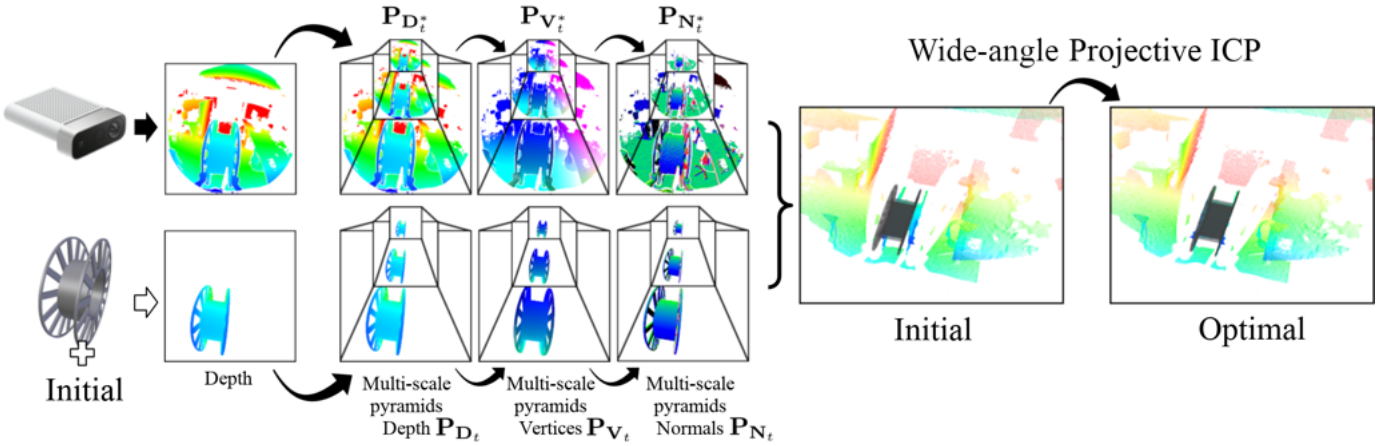


Fig. 4: Process flow of the 3D object tracking with its CAD model as input within a wide-angle depth image captured with a Microsoft Azure Kinect. Before the optimization runs, the 3D model shows shifted w.r.t. the captured data (see "Initial") whereas both perfectly align after optimization (see "Optimal").

experiments of this paper, the control framework relies solely on the visual estimated information and force sensing to make decisions based on: (i) robot's localization within vSLAM map or based on visual tracking ; (ii) current target waypoint expressed within vSLAM map; (iii) object localization w.r.t the robot using embedded tracking; and (iv) force sensors data in the feet and grippers.

C. Enriched vSLAM map

1) Offline map enrichment: As mentioned previously, our work focuses on closed-environments of manufacturing industries. The surrounding main buildings and manufacturing implements, such as automation machines, are known and have low instant appearance variability. Therefore, we can use vSLAM in two distinct steps [1]: (i) the creation of the environment map, and (ii) the use of this map to localize the robot. The map creation is done by manually handling the camera (the same model as the one embedded on the robot) in the environment. The map is a 3D point cloud representing the ground and the surroundings of the robot start pose and of the object start and targeted poses. The ground is the horizontal plane that is registered with the map. It is used within the presented framework (Sec. III-B) to compute the desired footsteps. The robot start pose does not have to be set accurately because the robot is able to localize itself within the map at the beginning of the deployment. When the manipulated object must reach accurately a pose defined with respect to an industrial machine, as in our most challenging experiments where the loco-manipulation of a large object must end by inserting the object in a machine (see Sec. VI-D and VI-C), we first perform the 3D registration of the machine 3D model with the 3D point cloud of the map. By doing so, the object targeted pose is straightforwardly transformed from the industrial machine coordinate system to the one of the map, together with a pre-defined set of waypoints (e.g., approach waypoint). They define the path to follow by the manipulated object in the coordinate system of the map. We call an *enriched map* the union of a map, pre-defined waypoints and pre-registered poses.

2) Online temporary map enrichment: The humanoid robot uses the enriched map for vSLAM-based localization within the environment. However, the estimated pose is not always accurate when vSLAM is used in localization only mode. Actually, the latter localization relies on the map structure made of keyframes. Despite computationally efficient, such localization scheme is only accurate from similar viewpoints than those the camera had during the map creation. To counterbalance this drawback, the vSLAM-based localization can run with the mapping mode to continuously add information to the existing map. This is the dual idea of [4] where previous maps are used to improve the current localization while mapping, whereas is our work, we propose to map while localizing with respect to a previous map. The difference is subtle but effective and since [4] could observe an improvement of the localization accuracy in their case, we expect an improvement too.

However, mapping while manipulating the large object adds parts of it to the map several times because the manipulated object is obviously in the camera field-of-view. This creates artifacts in the map, resulting in poor localization robustness. To solve this issue and keep the localization accuracy by mapping while localizing within a pre-built map, we combine the visual tracking of the object with the vSLAM to *hide* the manipulated object to the vSLAM process. Hiding the object for vSLAM is done by dynamically masking it, since its location in every captured image is known from online tracking. Hence, the environment map is also enriched online safely during loco-manipulation to ensure the accuracy. But, contrary to the offline enrichment, the online enrichment is forgotten once the manipulated object reaches the targeted location in order to keep a lightweight memory of the working environment. The results of this approach are discussed with comparisons in Sec. VI-B.

IV. Object Tracking

Robot localization from vSLAM is not enough to grasp and move an object to a desired pose. The estimation of the manipulated object is required to meet at least two needs: (i) determining the next action by the planning/control

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.

framework, and (ii) to achieve object manipulation and re-grasping sequence, robustly, during the action. In this section, we discuss our approach for object tracking based on virtual visual servoing [51]. Figure 4 illustrates the process flow from the captured and rendered input depth frames to the output optimal pose that aligns the best the rendered object (right of Fig. 4, gray object) to the captured data (right of Fig. 4, colorful point cloud).

Define the transformation between object o and camera c,

$$ {}^{c}\mathbf{T}_{o} = \begin{pmatrix} {}^{c}\mathbf{R}_{o} & {}^{c}\mathbf{tr}_{o} \\ 0 & 1 \end{pmatrix} \in \mathrm{SE}(3), \tag{1} $$

with ${}^{c}\mathbf{R}_{o} \in \mathrm{SO}(3)$ is the rotation component, and ${}^{c}\mathbf{tr}_{o} \in \mathbb{R}^3$ is the translation component. The goal of the tracking process is to estimate ${}^{c}\mathbf{T}_{o}$ at any time t. During loco-manipulation, both the manipulated object and the camera move but to estimate their relative transformation, one can either estimate updates of the object pose with respect to the camera or, dually, updates of the camera pose with respect to the object. Without loss of generality we consider the latter, leading to writing the dependence to t by ${}^{c_t}\mathbf{T}_{o}$, so that:

$$ {}^{c_t}\mathbf{T}_{o} = {}^{c_t}\mathbf{T}_{c_{t-1}} {}^{c_{t-1}}\mathbf{T}_{o} \tag{2} $$

where ${}^{c_{t-1}}\mathbf{T}_{o}$ is the estimated object pose at time $t-1$. Thus, for each captured wide-angle depth-image $\mathbf{D}_t^*$, ${}^{c_t}\mathbf{T}_{c_{t-1}}$ is the solution of an optimization problem aligning (or registering) the wide-angle depth-image $\mathbf{D}_t$ of the 3D object, rendered with ${}^{c_{t-1}}\mathbf{T}_{o}$, to the desired $\mathbf{D}_t^*$.

First, the captured wide-angle depth-image $\mathbf{D}_t^*$ is obtained at time t. $\mathbf{D}_t^*(\mathbf{u}) \in \mathbb{R}_+$ is the depth measure in mm, and $\mathbf{u} = (u, v)^\top \in \mathrm{U} \subset \mathbb{R}_+^2$ is the pixel coordinates in the wide-angle depth-image domain U. When nothing is captured within the range of the wide-angle depth sensor, or when $\mathbf{D}_t^*(\mathbf{u})$ is not defined in U because the wide-angle depth-image is an ellipse in a square image, or due to some material properties, $\mathbf{D}_t^*$ is depthless at $\mathbf{u}$ (see Fig. 4, top row, *Depth* column). This is coded as $\mathbf{D}_t^*(\mathbf{u}) = 0$.

Second, the other wide-angle depth-image $\mathbf{D}_t$ is obtained from the object 3D CAD model o rendered from a virtual camera pose ${}^{v}\mathbf{T}_{o} = {}^{c_{t-1}}\mathbf{T}_{o}$. In order to have $\mathbf{D}_t$ comparable with $\mathbf{D}_t^*$, the projection function Φ (projection and distortion models [52]) of the virtual camera v uses the intrinsic parameters of the actual Depth camera (see Fig. 4, bottom row, *Depth* column). Indeed, one of the main novelty of this work is to run the data association stage in the captured image plane, featuring strong distorsions, contrary to the previous works that undistort the captured depth image as a prior. This key difference prevents (i) a strong decrease of the resolution at the center of the image (where a manipulated object is usually) due to the undistortion process, if the output image resolution is the same as the input one, and (ii) a strong increase of the image resolution, if avoiding a decrease of resolution at the image center, due to the very large field-of-view. The former (i) would obviously decrease the accuracy of the pose estimation, a major issue for further grasping stage, whereas (ii) would increase the processing time due to a larger amount of pixels to process.

But the high distorsions inherent to the wide-angle image imply using a high resolution mesh for the object 3D CAD model to obtain $\mathbf{D}_t$ without geometric artefacts. Indeed, with such distorsions a scene straight line appears as a curve in the image. Hence, the 3D CAD model of the object, which has been designed classically with the least number of faces and vertices, is subdivided off-line[3]. The increased density of vertices and faces in the 3D mesh enables the deformation of its projection in the image plane according to the distortion parameters of the camera.

We can reasonably assume the inter-frame motion is not large between two successive acquisitions of $\mathbf{D}_t^*$ because the object is manipulated by the robot that straightforwardly imposes a dynamics suitable to verify this assumption. But in order to extend the fast data association of the projective Iterative Closest Point (ICP) algorithm [53] to wide-angle depth-images the alignment algorithm considers multi-scale pyramids, toward a very computationally efficient implementation compared to searching at each iteration for closest neighbors in 3D space. $\mathbf{D}_t^*$ itself is represented as a multi-scale pyramid noted $\mathbf{P}_{\mathbf{D}_t^*}$ but we also compute the multi-scale pyramid $\mathbf{P}_{\mathbf{V}_t^*}$ of the 3D vertices by back-projecting $\mathbf{D}_t^*$ thanks to the intrinsic parameters of the camera as well as $\mathbf{P}_{\mathbf{N}_t^*}$, the multi-scale pyramid of the normals to the object surface at each 3D vertex. Figure 4 (top row, *Multi-scale pyramids Depth*, *Vertices* and *Normals*) illustrates these three multi-scale pyramids. As $\mathbf{D}_t$ is obtained using the parameters of the projection and the distortion models that apply to $\mathbf{D}_t^*$, the multi-scale pyramids $\mathbf{P}_{\mathbf{D}_t}$, $\mathbf{P}_{\mathbf{V}_t}$ and $\mathbf{P}_{\mathbf{N}_t}$ are computed similarly but from $\mathbf{D}_t$. One must note that for both $\mathbf{P}_{\mathbf{N}_t^*}$ and $\mathbf{P}_{\mathbf{N}_t}$, the normals are computed from the vertices in $\mathbf{P}_{\mathbf{V}_t^*}$ and $\mathbf{P}_{\mathbf{V}_t}$ instead of using normals that could come from the object 3D CAD model (as in [54]). By doing so, we easily enforce corresponding normals $\mathbf{N}_t^*$ and $\mathbf{N}_t$ to point toward the same direction.

The alignment optimization process runs iteratively from the coarsest to the finest pyramid level and relies on dense depth pixel-to-pixel correspondances $(\mathbf{u}, \hat{\mathbf{u}})$, $\hat{\mathbf{u}} \in \mathbb{R}_+^2$ being the corresponding pixel in $\mathbf{P}_{\mathbf{D}_t^*}$ of $\mathbf{u}$, noted $\mathbf{P}_{\mathbf{D}_t^*}(\hat{\mathbf{u}})$. To obtain the correspondances, the current guess ${}^{c}\tilde{\mathbf{T}}_{c_{t-1}}$ of the transformation matrix to align $\mathbf{P}_{\mathbf{D}_t}$ to $\mathbf{P}_{\mathbf{D}_t^*}$, first transforms each vertex $\mathbf{P}_{\mathbf{V}_t}(\mathbf{u})$ observed at coordinates $\mathbf{u} \in \mathrm{U}$ in the depth-image $\mathbf{P}_{\mathbf{D}_t}$. Then, each transformed vertex is projected at $\hat{\mathbf{u}} = \Phi({}^{c}\tilde{\mathbf{T}}_{c_{t-1}}, \mathbf{P}_{\mathbf{V}_t}(\mathbf{u}))$ in $\mathbf{P}_{\mathbf{D}_t^*}$.

But because the captured depth-image may suffer of depthless pixels and we want to avoid mismatches, not all corresponding pairs $(\mathbf{u}, \hat{\mathbf{u}})$ are considered to solve the optimization problem. Only the subset $\Omega \subseteq \mathrm{U}$ of pixel coordinates $\mathbf{u}$ is considered, following these criteria:

$$ \begin{aligned} \Omega = \quad & \{\mathbf{u} \in \mathrm{U} : \mathbf{D}_t(\mathbf{u}) > 0, \text{ and } \mathbf{D}_t^*(\hat{\mathbf{u}}) > 0, \\ & \text{and } ||{}^{c_t}\tilde{\mathbf{T}}_{c_{t-1}} \mathbf{P}_{\mathbf{V}_t}(\mathbf{u}) - \mathbf{P}_{\mathbf{V}_t^*}(\hat{\mathbf{u}})||_2 \leq \delta_d, \\ & \text{and } \tilde{\mathbf{R}} \mathbf{P}_{\mathbf{N}_t}(\mathbf{u}) \times \mathbf{P}_{\mathbf{N}_t^*}(\hat{\mathbf{u}}) \leq \delta_\theta \}, \end{aligned} \tag{3} $$

where $\tilde{\mathbf{R}}$ is the rotation part of ${}^{c_t}\tilde{\mathbf{T}}_{c_{t-1}}$, δ_d is a threshold in meter and δ_θ is a threshold in radian, experimentally chosen as the best trade-off maximizing the number of pixels $\mathbf{u}$ considered in Ω while rejecting outliers that should not be considered in the energy to minimize.

[3]Subdividing a mesh is very classical, here done with Blender's (https://www.blender.org) built-in subdivide operation preserving the original shape.

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.

The optimization process for alignment is expressed leveraging the point-plane metric [55] as the following energy:

$$\mathbf{E}({}^{c}\tilde{\mathbf{T}}_{c_{t-1}}) = \sum_{\mathbf{u}\in\Omega} ||({}^{c_t}\tilde{\mathbf{T}}_{c_{t-1}}\mathbf{P}_{\mathbf{V}_t}(\mathbf{u}) - \mathbf{P}_{\mathbf{V}_t^*}(\hat{\mathbf{u}}))^{\top}\mathbf{P}_{\mathbf{N}_t^*}(\hat{\mathbf{u}})||_2, \tag{4}$$

to be minimized exploiting the Cholesky decomposition [54].

At convergence, the tracking returns ${}^{c_t}\mathbf{T}_o$, which is the estimated pose of the manipulated large-scale object w.r.t. the robot embedded camera frame.

The tracking is integrated as a component of the control framework so as to estimate ${}^{w}\mathbf{T}_o$ the object pose w.r.t. the environment using ${}^{c_t}\mathbf{T}_o$ and the camera pose in the environment map ${}^{w}\mathbf{T}_{c_t}$, with vSLAM (see Sec. III-C). Yet, recall that obtaining reliable ${}^{w}\mathbf{T}_{c_t}$ estimates requires a mask computed from the object 3D model projected from pose ${}^{c_t}\mathbf{T}_o$ in the captured depth-image, so that vSLAM ignores the manipulated object. We do not create a mask *per se* but we basically set $\mathbf{D}_t(\mathbf{u})$ to 0 if $\mathbf{u} \in \mathrm{S}$; where S is the set of pixels belonging to the object on the camera plane after the projection step.

Then we rectify $\mathbf{D}_t$ to match the RGB frame intrinsic properties. Finally, we give as inputs this rectified depth and the RGB frames to the vSLAM that considers a pixel from depth and RGB frames only if the depth information exists.

For the robot motion planning and control, ${}^{w}\mathbf{T}_o$ is transformed to ensure physical constraints depending on the object type. For instance, for both the bobbin and wheelbarrow objects considered in the loco-manipulation demonstrations of this article (see Sec. V), ${}^{w}\mathbf{T}_o$ is transformed such that the object rolling axis is horizontal.

V. Experimental Setup

We assess our approach through experiments using two different Kawada Robotics humanoid robots: HRP-2KAI and HRP-5P in wheelbarrow (Fig. 10) and bobbin loco-manipulation (Fig. 11 and 14) use-cases. We used a commercially available wheelbarrow of size $170 \times 68 \times 66$ cm. Its weight is approximately 12 kg and its 3D CAD model is shown in Fig 5. The bobbin is a rolling object of 133 cm diameter, thick of 40 cm, found in many large-scale manufacturing industries and contains rolled-up materials of different kind, such as cables or bands, and are generally heavy (those we study weigh approximately 140 kg). Figure 5 shows the CAD model of the bobbin that is rather ideal compared to the used bobbin in experiments: it has been used for years in a plant, thus it is deformed here and there. Interestingly, this fact will show the robustness of the tracker described in Sec. IV.

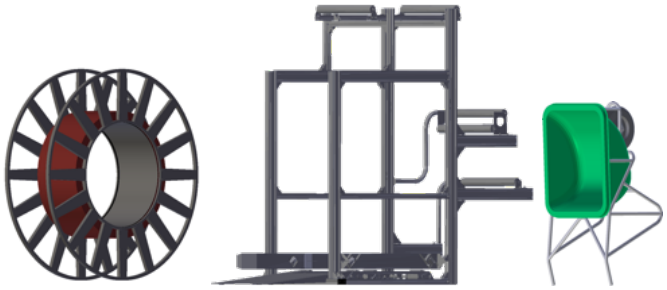


Fig. 5: 3D model of the bobbin, a simplified unwinder and a wheelbarrow. The bobbin and unwinder (real and models) are provided by an industrial partner.

Bobbins are inserted in winder/unwinder automation systems to be filled/unfilled and removed after. In our experiments, we use a mock-up of an industrial unwinder (see its CAD model in Fig 5) that is 50 cm width. The lateral insertion tolerance error is about ± 5 cm max, if the bobbin orientation error is zero, and a maximum of $\pm 4.4°$ of orientation tolerance, if the bobbin lateral position is perfect.

Both robots are controlled using the task-space framework `mc_rtc` (Sec. III-B). Only the manipulated object and precomputed values are changed (e.g., grasping poses).

We mounted on both robots an Azure Kinect camera using a 3D printed attachment. Azure Kinect camera integrates a 1-Megapixel Time-of-Flight depth camera [10] which offers an operating angle (or FoV) of $120° \times 120°$, while the RGB sensor one is of $90° \times 59°$. In comparison, the Kinect V2 has an operating FoV of $70.6° \times 60°$ for the depth sensor and $84.1° \times 53.8°$ for the RGB one; the Asus Xtion Pro Live (Asus) has an operating range of $58° \times 45°$ for both sensors. Note that the minimal distance required to perceive a large object is: 0.25 m for the Azure, 0.5 m for Kinect V2, and 0.8 m for the Asus. From these specifications, the Azure appears the best candidate to fully perceive an object at a close range without necessarily pointing at it. It is also the only one capable to acquire a wide-angle depth image. In our experiments, we set the depth camera acquisition rate to 30 Hz, implying a 512×512 pixels resolution (Wide Field-Of-View, WFOV, 2×2 binned mode). The camera-robot extrinsic parameters are obtained via existing calibration tools [56].

On an implementation level for the object tracking component, we render the objects 3D model with `OpenGL`. Then the multi-scale pyramids are computed also with `OpenGL` while the optimization extends the `CUDA`-based implementation of the ICP (`ICPCUDA`[4]) to using the Brown-Conrady distortion model [52], considered to model the depth-images captured with the Azure Kinect camera. `OpenGL` and `CUDA` data are shared thanks to the availability of interoperability functions. Lastly, the pose estimation is performed on CPU with `Eigen` thus we only copy a 6×6 matrix and a 6×1 vector from device (`CUDA`) to host (CPU). It allows us to have real-time capability to track the desired object with an Azure Kinect camera. We run the tracking on a laptop equipped with a GeForce RTX 2060 NVidia GPU. Regarding the thresholds of the data association step of the tracker (Sec. IV), they are empirically set to $\delta_d = 0.05$ m and $\delta_\theta = 45°$, once for all.

The other thresholds of our approach are those to decide if the manipulated object is facing or has reached a waypoint: Δ_d and Δ_θ (see Sec. III-B). They are set to $\Delta_d = 0.12$ m and $\Delta_\theta = 10°$, if the next object motion task is of the same type as the current (e.g., a succession of `StraightObject` forward) or $\Delta_d = 0.08$ m and $\Delta_\theta = 7.5°$, if the next task is different (e.g., `StraightObject` followed by `TurnObjectFacing`). Again, the latter values are set empirically but once for all the experiments reported in the article. Having two pairs of $(\Delta_d, \Delta_\theta)$ reveals sufficient to

[4]https://github.com/mp3guy/ICPCUDA

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.

avoid unnecessarily switches between object straight and turn motions.

RTAB-Map is the vSLAM used with the originality to feed it with images from which the manipulated object is dynamically hidden. We created a map of the experimental environment, an indoor laboratory, prior to running the experiments. We also performed the registration of the unwinder, w.r.t. which the target and approach poses are defined, within the latter map to obtain the enriched map (Sec. III).

With these elements we performed various experiments to assess our approach that we discuss together with the obtained results in the next section. A multi-media video attached to this article synthesizes these experiments.

VI. Experimental Results

In this section we will discuss the experimental results concerning the object-tracking, the vSLAM, and the wheelbarrow and bobbin manipulations use-cases by the humanoid robots HRP-2KAI and HRP-5P.

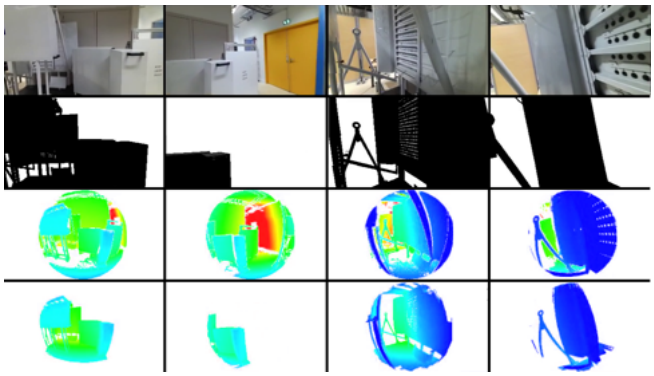
A. Visual tracking experiments

Our tracker is a carefully implemented adaptation of the projective ICP to wide-angle depth images directly. Since the seminal projective ICP has been extensively evaluated [54], this section mainly focuses on assessing qualitatively our visual tracking approach in its ability to track an object in three different situations.

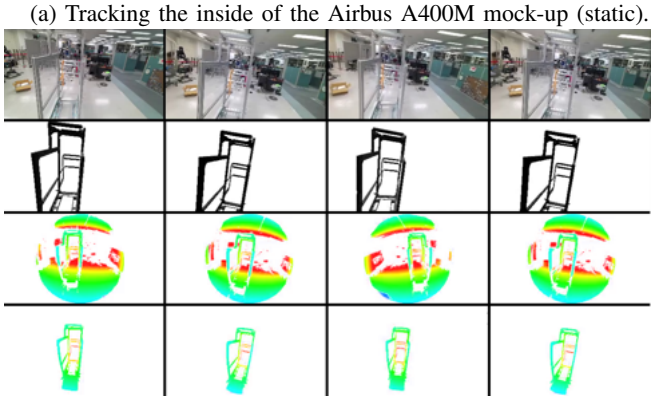
The first use-case is from aircraft manufacturing applications. A humanoid shall access a confined textureless mock-up where all vSLAM we tried failed to localize the robot [1]. Figure 6a shows four excerpts of our tracking in a sequence where the camera starts out of the mock-up and goes inside the mock-up. The tracking is successful as shown by the alignements of the 3D model projection in the depth frame (Fig. 6a, 4th row) with the captured depth image (Fig. 6a, 3rd row). In this setup, the estimated pose provides in fact an estimation of the robot localization within the mock-up.

Second, we assess the tracking of another static object in the environment: the unwinder (Fig. 6b). Contrary to the previous use-case, the camera moves around the object that is smaller and composed of thinner metallic bars comparing to the plates of the aircraft mock-up. Though smaller than the unwinder, the wheelbarrow is also successfully tracked too (Fig. 6c).

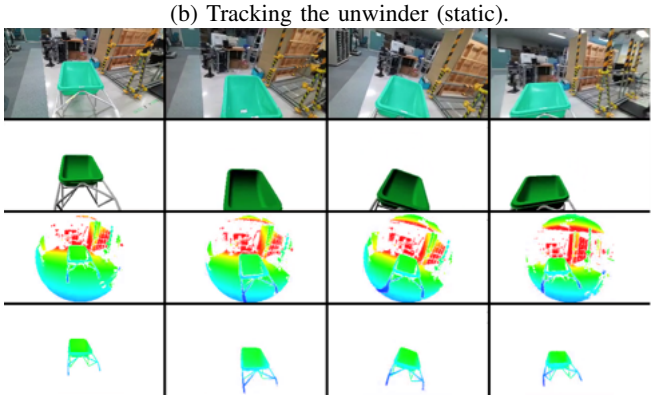
Third, the hardest case is considered: tracking a manipulated object while the camera is moving with the robot. Contrary to the tracking of static objects, this case features motion in images that combines both object and camera motions. Moreover, the most challenging object to track in our collection of use-cases is the bobbin as it must be manipulated to make it roll by using alternatively the left and the right hands that partially occlude the bobbin edges. Despite these challenges, the tracking is successful for quite a long sequence in our 5 m experimental space (Fig. 6d). This qualitative evaluation is done with the wide-angle depth camera mounted on the head of a human manipulating the bobbin to show the tracker has no clue of any control inputs or other sensor inputs and is truly vision-based only.



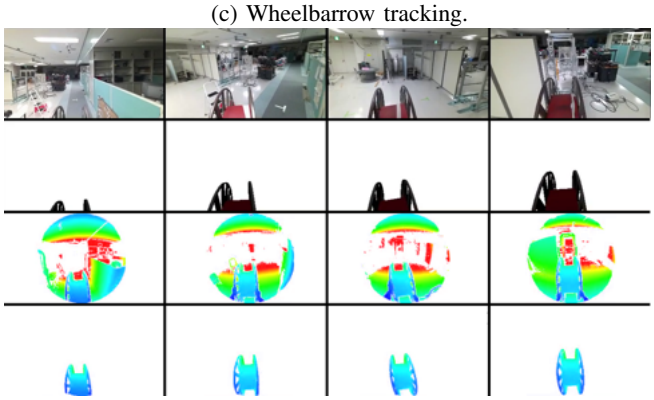
(a) Tracking the inside of the Airbus A400M mock-up (static).



(b) Tracking the unwinder (static).



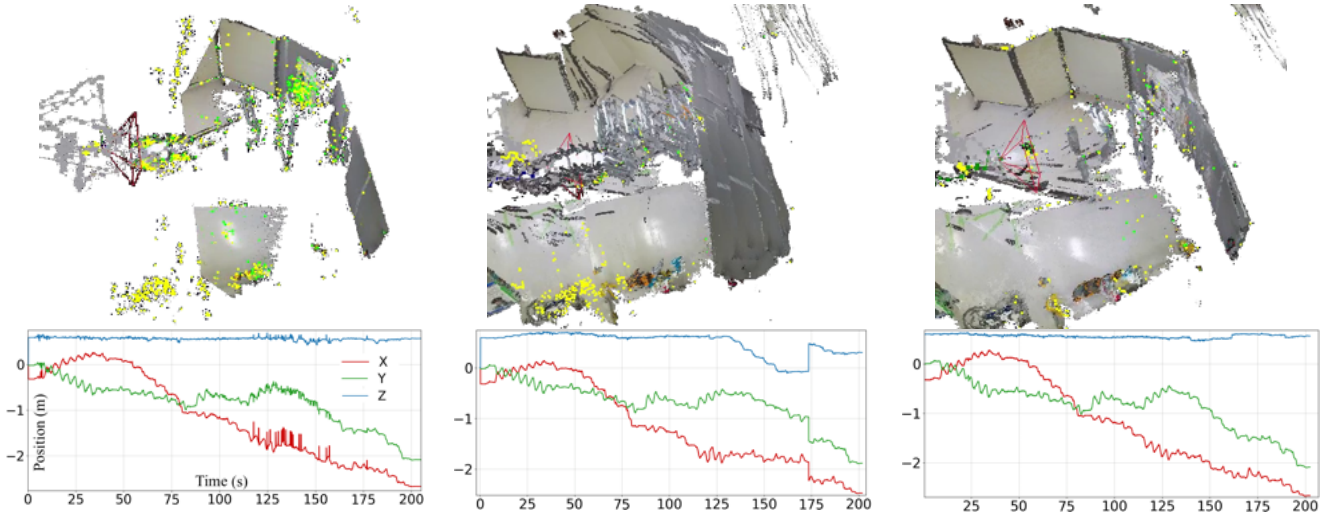
(c) Wheelbarrow tracking.



(d) Tracking the bobbin manipulated by an operator.

Fig. 6: Rows in (a-d): 1st, captured RGB images; 2nd, 3D model at optimal poses rendered in the RGB image frame (not used for tracking); 3rd, captured wide-angle depth frames; 4th, computed depth-images of the 3D model at optimal poses.

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.



(a) Localization in a pre-built map

(b) Mapping while localizing in the pre-built map

(c) Masking the manipulated object for mapping while localizing in the pre-built map

Fig. 7: Impact of the online map enrichment on the localization relevance and stability. X, Y, Z are components of the translation of the estimated pose of the camera within the map: (c) and (b) are to be compared to (a).

Because of its fast computation for a single object, our tracker can also be run several times in parallel to track several objects in the captured depth-images. Yet, the maximum number of objects depends on the computation time used for each object; which is itself depending on the tracker implementation (see Sec. V). As an example, Fig. 8 shows a snapshot of the successful simultaneous tracking of the manipulated bobbin and the static unwinder, from a record of an experiment done with HRP-5P (detailed in Sec. VI-D).

However, being able to track the unwinder is not enough to perform bobbin loco-manipulation experiments: the unwinder is not always in the FoV nor in the range of the camera depth sensor. Thus, it's known pose in the enriched map for the vSLAM is necessary, and actually will reveal sufficient to successfully achieve the bobbin insertion in the unwinder.

B. vSLAM

During a bobbin loco-manipulation experiment we recorded data from the camera and performed the analysis *a posteriori*.

First, we applied the vSLAM in localization mode only. As in this case the estimation heavily depends of the keyframes existing in the map, the estimated position is noisy when the current camera is far from the keyframes of the map (Fig. 7a), thus impacting the accuracy of robot pose estimation. This confirms the motivation of the strategy of mapping while localizing in a pre-built map (see Sec. III-C2).

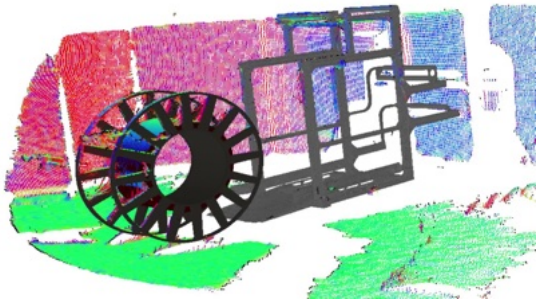


Fig. 8: Example of multi-object tracking: bobbin and unwinder. The colors illustrate the normals estimated from the captured point cloud. The grey parts correspond to the models.

But in our challenging case of localization while manipulating, the basic activation of mapping while localizing w.r.t. the pre-built map leads to the addition of parts of the manipulated object to the map (Fig. 7b), which is undesirable and leads to erroneous camera pose estimation: see the jump in estimated position around 175 s at the bottom of Fig. 7b, whereas the true camera does not jump at all, and the 0.5 m drift in estimated Z translation from 125 s to 150 s whereas the camera remains at a constant altitude in that period of time.

Finally, the result of the manipulated object tracking is used to ignore its corresponding part in the RGB-D data (Fig. 9b) set as input of the vSLAM achieving the mapping while localizing w.r.t. to a pre-built map. This time, the final map is not erroneous and does not include any part of the manipulated object (Fig. 7c). As a positive side effect, the final map is also much denser. We do not exploit this characteristics further but the main result is the estimated positions are much smoother, without drift nor sudden jump as Fig. 7c (bottom) shows.

This result highlights the contribution on the visual perception in this work and validates the combined visual tracking and SLAM process to localize the robot in its environment while the object is loco-manipulated toward its target pose.

C. Experiments with HRP-2KAI humanoid

1) Loco-manipulation of wheelbarrow: The HRP-2KAI humanoid robot is set to manipulate a wheelbarrow. In this experiment, the operator uses `mc_rtc`'s Graphical User Interface to program high level actions for the mission. First, the operator selects `WalkRelative` to make the robot walk to the wheelbarrow grasping pose by means of visual tracking, see Fig. 10(a). Then, the user added the `GraspObject` task

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.

for the robot to grasp the wheelbarrow with its grippers and to lift it up followed by the wheelbarrow loco-manipulation. The result of this action is shown in Fig. 10(b). Once the robot lifts the wheelbarrow, successive `MoveObject` actions move forward of 0.5 m, turn the wheelbarrow in place of 30° then move forward by 0.5 m are given; see Fig. 10(c), and the accompanying video. HRP-2KAI successfully brings the wheelbarrow to the operator chosen targets.

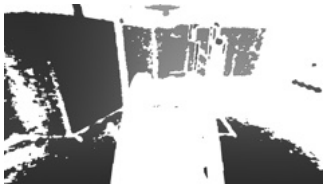
2) Loco-manipulation of a bobbin: In the bobbin loco-manipulation case, several waypoints were pre-defined thus `MoveObject` actions are automatically triggered without any operator input. We describe in depth the results obtained with HRP-2KAI for the loco-manipulation of the bobbin (Fig. 11). Target grasping poses of the bobbin are pre-defined in its own frame in-between spokes. During the loco-manipulation, due to the bobbin inertia and friction with the ground, there is a possibility that it did not reach the desired pose computed for the `MoveObject` action (Sec. III). Object visual tracking compensates for this discrepancy by providing the object configuration w.r.t. the robot. Thus, the closest grasping pose target is obtained straightforwardly to feed the `GraspObject` action. In the experiment reported in Fig. 11, as for many other trials, HRP-2KAI did not miss any grasp.

Figure 12 shows the align angle between the bobbin main axis and the waypoint one. The `MoveObject` actions are illustrated in Fig. 12 for HRP-2KAI experiment: red areas (top) correspond to the forward action while the blue ones to the turn action. Around 280 s, the estimated distance (thanks to the object-tracking and vSLAM) between the bobbin and the waypoint falls below the pre-defined threshold Δ_d (see Sec. III-B and V) for the first waypoint. At 295 s, during 15 s where the object is not moving, the robot performs `WalkRelative` action to a pre-defined pose w.r.t. the object. Meanwhile, the estimated angle between the object and the waypoint shrinks –indeed the robot is turning the object.

In order to turn the object, due to its kinematics, HRP-2KAI has to stand on the side of the bobbin to swing its center of mass (CoM) during the action. In addition, HRP-2KAI lowers its CoM during that swing in order to avoid stretching the leg to a kinematic limit. During these motions, the visual tracking does not fail despite big changes of the camera viewpoint close to the (big) bobbin, so does the vSLAM.

For any types of `MoveObject` actions, the monitored value (angle or distance) reduces up to a flat part (e.g., around 240 s). During this time, the robot is performing a re-grasping sequence or it is walking relatively to the object.

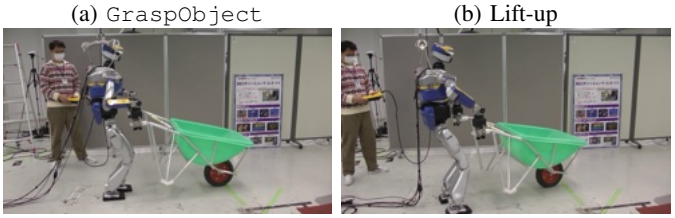
(a) Wide-angle depth image



(b) Rectified depth image with dynamic mask

Fig. 9: (a) is the input to object tracking and (b) is the output rectified masked depth for vSLAM

(a) `GraspObject` (b) Lift-up



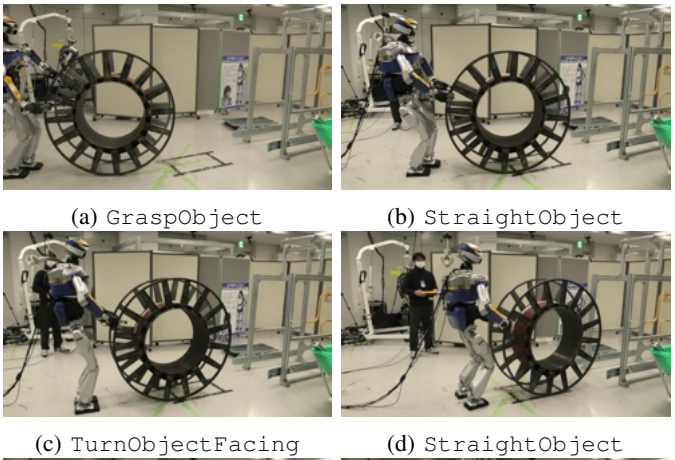
(c) `StraightObject` (d) `TurnObjectAlign`

Fig. 10: HRP-2KAI's experiment moving the wheelbarrow.

Thus, the bobbin is not moving, which is correctly estimated (see the nearly flat red curve at 240 s in Fig. 12).

The last waypoint is in the middle of the unwinder, thus around 450 s (i.e., by the end of the experiment) the bobbin is within the unwinder as shown in Figure 11 (f).

The angle shown in Fig. 13 corresponds to the facing angle. Recall it is the angle between the bobbin forward axis and the waypoint position (see Sec. III-B). As the distance between the object and the waypoint reduces (during the forward action), the bobbin-waypoint vector becomes more and more sensitive to estimation noise. Around 275 s in Fig. 13, when the bobbin is close to the waypoint, the framework monitors the angle to align the bobbin w.r.t. the waypoint. Once it is done, at 320 s the bobbin is within the thresholds for the distance Δ_d (see Fig. 12) and for the facing angle Δ_θ (see Fig. 13). Hence, the



(a) `GraspObject` (b) `StraightObject`

(c) `TurnObjectFacing` (d) `StraightObject`

(e) `StraightObject` (f) `ReleaseObject`

Fig. 11: Motion actions for the HRP-2KAI bobbin experiment.

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.

framework switches to the next forward action. Around 350 s, the facing angle is moving away the threshold; it happens during the forward action itself. This correctly estimated the true drift which is due to the imperfect bobbin that has been used in a plant for years. So, it appears that the bobbin is within the threshold distance, and the framework decides (again) to align the object with the waypoint.

By repeating these steps, illustrated in Fig. 11, HRP-2KAI can successfully bring the bobbin within the unwinder, without ramming into it, and from different initial locations, hence highlighting the *high efficiency of the whole loco-manipulation* process, from the visual tracking to the motion control (recall the lateral tolerance is ± 5 cm and the orientation tolerance is $\pm 4°$, see Sec. V).

D. Experiment with the HRP-5P humanoid

HRP-5P is an updated version of HRP-2KAI that can be torque- or high-gain kinematic controlled. The reason behind putting additional effort in trying the same experiment with another humanoid are twofold: (i) assess the portability of our control strategy and our framework, (ii) have insight through comparative hardware on the hardware resilience requirements. Indeed, none of HRP-2KAI and HRP-5P has been designed to drive bobbins because they are general purpose robots. However, HRP-5P is designed to be deployed in large construction sites [57] and could reveal interesting in industrial tasks at shipyards, airplane factory or for dismantling nuclear power plants. Compared to HRP-2KAI, HRP-5P comes with additional degrees-of-freedom (DoFs) in the waist and in each arm; it has more freedom in its motions and it is closer in emulating human motions. In addition, on average, HRP-5P is two times more powerful than HRP-2KAI, i.e., it has more joint power to weight ratio.

Both robots have similar walking capabilities; however for the `WalkRelative` action, thanks to being taller and having longer arms, HRP-5P can stand further away from the bobbin. For `StraightObject`, by using only one arm HRP-5P can move the bobbin much further compared to HRP-2KAI (about 63.8 cm for HRP-5P vs. 28.4 cm for HRP-2KAI) and hence reducing the number of re-grasping sequence needed.

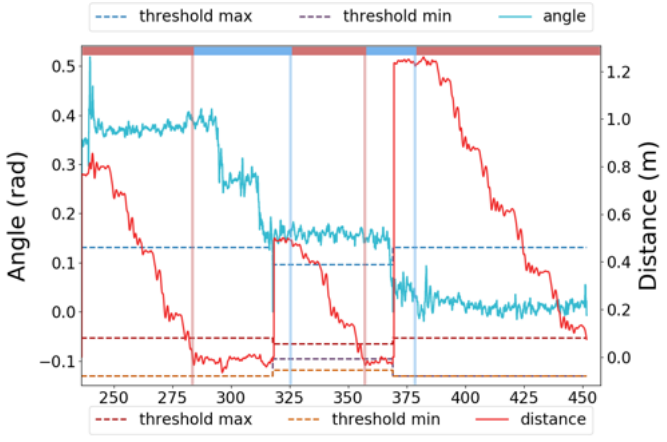


Fig. 12: Bobbin's angle (blue line) and distance (red line) relatively to the next waypoint pose (position for the distance, orientation for the angle) during `MoveObject` actions (HRP-2KAI experiment of Fig. 11). At the top axis, red areas correspond to forward and blue ones to turn motions.

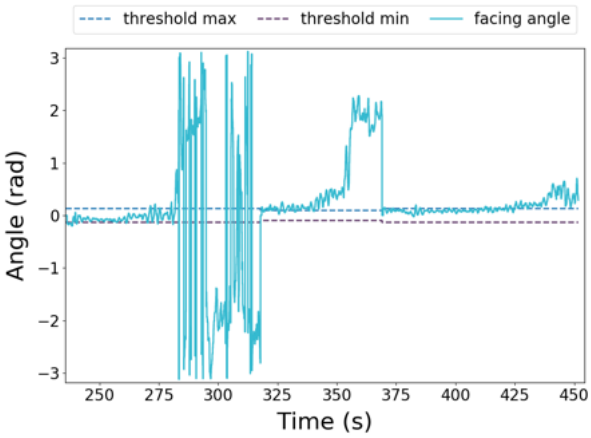


Fig. 13: Facing angle between waypoints (positions) and bobbin (forward axis) from HRP-2KAI experiment.

During `TurnObjectFacing/Align`, HRP-5P does not have to walk on the lateral side before initiating the action thanks to its additional DoFs and power (e.g., the additional DoF of the waist allows the Center of Mass to swing). This reduces the number of footsteps and time needed between the `MoveObject` transitions, so the behavior of HRP-5P is actually less challenging for the visual tracking than HRP-2KAI's. Also, during the bobbin rotation on the spot, HRP-5P's foot that is in contact with the ground does not slide as much as the one of HRP-2KAI; it reduces the number of time the robot has to re-align itself with the bobbin by executing the `WalkRelative` action. Hence, the error between the expected bobbin rotation and the rotation achieved during one action is smaller with HRP-5P than with HRP-2KAI.

All these differences between the two robots lead HRP-5P to achieve the whole loco-manipulation experiment, from the robot startup to the end of the `ReleaseObject` action after fully inserting the bobbin in the unwinder, in less than *half the time duration* required by HRP-2KAI ($3'15''$ for HRP-5P vs. $8'32''$ for HRP-2KAI for their fastest runs) to achieve the same experiment (only the start poses of the robot and the bobbin are of course not exactly the same). As for HRP-2KAI, during the whole experiment shown in Fig. 14 HRP-5P did not miss a grasp thanks to the object-tracking and successfully bring the bobbin inside the unwinder.

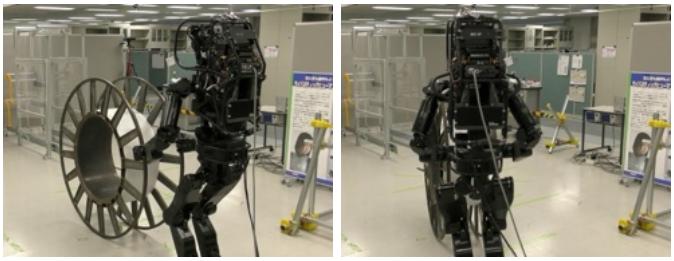
VII. Discussion

In this section, we discuss limitations and possible failures together with further research ideas to increase the performance of similar complex set-ups.

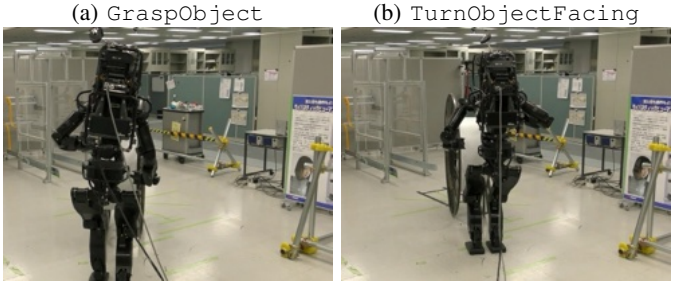
A. Limitations and failures

Regarding limitations, all experiments and developments have been made considering flat terrains. This is because all the manufacturing use-cases we are tackling have flat shop-floors. In terms of perception, i.e., vSLAM and visual tracking, and also in terms of whole-body control (i.e., `mc_rtc`) there

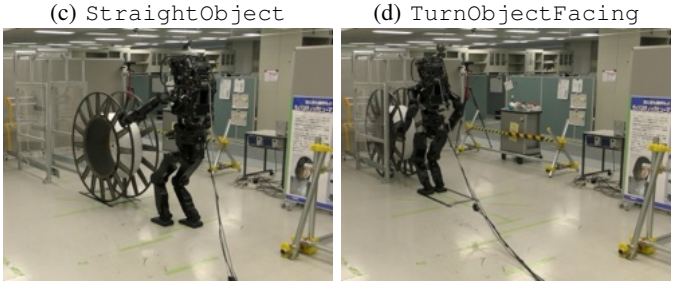
IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.



(a) `GraspObject` (b) `TurnObjectFacing`



(c) `StraightObject` (d) `TurnObjectFacing`



(e) `StraightObject` (f) `ReleaseObject`

Fig. 14: Motions during the HRP5-P experiment with bobbin

is no major issue that prohibits their use in uneven terrains. However, the planning in [24], [25] has to be thoroughly revisited. As another shortcoming previously mentioned in Sec. III-B, the waypoints are defined by the operators, however we may consider automatic generation of (milestones) waypoints (i.e., by following encoded safety rules or practices) that the operator can confirm or discard or edit at will. As for now, the planning framework is agnostic to human workers that may populate the shop-floor and no reactive strategy is implemented yet. Finally, the walking is not continuous but stops between every action. We explain in Section VII-B how we are planning to deal with this limitation.

Failures in the mission execution can occur in the case of: (i) a bad offline map recording, (ii) a bad registration of the target in the map, (iii) a wrong estimation within the map, (iv) noises in the visual estimation or (v) a bad choice in the thresholds Δ_d and Δ_θ to validate a successful waypoint reach. (i) and (ii) can even be related: due to (i) the unwinder mapping can be very partial, thus preventing an accurate registration of the unwinder 3D model to the pre-built map, hence leading to a poor enriched map. Furthermore, (i) and (iii) can also be related: when the robot starts at an initial pose around which the environment wasn't sufficiently mapped, the risk is to start creating online a new map instead of enriching (temporarily) the pre-built map.

An experimental solution to (i), (ii) and (iii) would be a very careful initial mapping (to make the pre-built map) around the robot start poses and of the unwinder: as the accompanying video shows, many other trials we conducted are successful.

More generally speaking, one can overcome (i), (ii) and (iii) by making several offline map recordings that can be run concurrently to have a stochastic estimate for the localization and registration. The best way to overcome (iv) is to duplicate the cameras and run parallel thread visual tracking. More importantly one can rely on the close-chain kinematics at contact to correct noises in the visual estimation.

B. Toward higher performances

As shown in the accompanying videos, the experiments are relatively slow w.r.t. human performances. In the HRP-5P's bobbin rolling experiments, each re-grasping takes roughly 4 s; during that time the bobbin is static. There are 21 changes of the gripper's configuration: in total, 1min24s ($21\text{s}\times4$) is spent only for regrasping. In brief, the regrasping takes 33% of the time of the experiment. In order to increase the execution speed, the following issues must be considered:

- The nature and disposition of the grippers is one of the sources of slowness. One needs to design appropriate robot grippers (and even the arm itself) to be able, for example, to slide on the bobbin handles while performing the loco-manipulation. We are designing a new gripper to be mounted with an orientation that lowers the unusual postures observed for the arms;
- Impacts are another limiting factor. Indeed, in order to proceed faster the robot shall be able to make contact with non-zero relative velocities, which we are prohibiting for the time being. Our recent results [58] allow extensions to our whole body control to handle impacts, yet it still has to be integrated to the `mc_rtc` framework.
- Lack of whole body preview: our task-space QP controller is local and hence agnostic to task-induced dynamics in future iterations (except for walking). Therefore, in order to go faster a flavor of model prediction shall be integrated to the whole-body controller.

VIII. Conclusion

In this paper, we presented a complete methodology that enables humanoid robots to operate and manipulate large-scale objects (e.g., a wheelbarrow and a bobbin), relying solely on its embedded sensors. The presented architecture has successfully demonstrated its capability to be precise enough to loco-manipulating a rolling bobbin following the requirements of a representative use-case provided by industries.

The success of our experiments relies on the improved vSLAM-based robot localization thanks to a dynamic masking of the large object manipulated by the humanoid. This also validates the interest of the recently introduced wide-angle depth, to track large-scale objects. Our future works aim at exploiting it fully for vSLAM too, as it is currently relying on rectified depth image.

Dense vSLAM turns to be very useful to align accurately the CAD model of industrial machines (e.g., the unwinder in which bobbins are inserted or removed, thus straightforwardly transferring insertion and approach poses of the bobbin, defined with respect to the unwinder by default). In future works,

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.

close to targets poses and waypoints could be automatically suggested to operators based on dense map data to scale from an environment of a workshop size to the entire factory.

On the more industrial side, our future efforts are also focused on (i) a complete unwinding task that requires tracking and manipulating deformable or flexible materials (namely those rolled around the bobbins), (ii) to accelerate, up to reaching human speed, both loco-manipulation use-cases, and (iii) dealing with the limitations stated in Section VII. It is only by tackling such issues that humanoid robots will reach the maturity for real-industry deployment perspectives.

REFERENCES

[1] A. Kheddar, S. Caron, P. Gergondet, A. Comport, A. Tanguy, C. Ott, B. Henze, G. Mesesan, J. Englsberger, M. A. Roa, P.-B. Wieber, F. Chaumette, F. Spindler, G. Oriolo, L. Lanari, A. Escande, K. Chappellet, F. Kanehiro, and P. Rabaté, "Humanoid robots in aircraft manufacturing: The airbus use cases," *IEEE Robotics Automation Magazine*, vol. 26, no. 4, pp. 30–45, December 2019.

[2] K. Bouyarmane and A. Kheddar, "Humanoid robot locomotion and manipulation step planning," *Advanced Robotics*, vol. 26, no. 10, pp. 1099–1126, 2012.

[3] S. Huang and G. Dissanayake, "A critique of current developments in simultaneous localization and mapping," *International Journal of Advanced Robotic Systems*, vol. 13, 10 2016.

[4] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[5] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. Leonard, "Past,present,and future of simultaneous localization and mapping: Towards the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[6] A. Paolillo, P. Gergondet, A. Cherubini, M. Vendittelli, and A. Kheddar, "Autonomous car driving by a humanoid robot," *Journal of Field Robotics*, vol. 35, no. 2, pp. 169–186, Mar. 2018.

[7] K. Chappellet, G. Caron, F. Kanehiro, K. Sakurada, and A. Kheddar, "Benchmarking cameras for OpenVSLAM indoors," in *International Conference on Pattern Recognition*, 2021, pp. 4857–4864.

[8] A. Tanguy, P. Gergondet, A. I. Comport, and A. Kheddar, "Closed-loop RGB-D SLAM multi-contact control for humanoid robots," in *IEEE/SICE Int. Symposium on System Integration*, 2016, pp. 51–57.

[9] I. Ballester, A. Fontán, J. Civera, K. H. Strobl, and R. Triebel, "DOT: Dynamic object tracking for visual SLAM," in *IEEE Int. Conf. on Robotics and Automation*, 2021, pp. 11 705–11 711.

[10] C. S. Bamji, S. Mehta, B. Thompson, T. Elkhatib, S. Wurster, O. Akkaya, A. Payne, J. Godbaz, M. Fenton, V. Rajasekaran, L. Prather, S. Nagaraja, V. Mogallapu, D. Snow, R. McCauley, M. Mukadam, I. Agi, S. McCarthy, Z. Xu, T. Perry, W. Qian, V. Chan, P. Adepu, G. Ali, M. Ahmed, A. Mukherjee, S. Nayak, D. Gampell, S. Acharya, L. Kordus, and P. O'Connor, "IMpixel 65nm BSI 320MHz demodulated TOF image sensor with 3micro-m global shutter pixels and analog binning," in *IEEE Int. Solid-State Circuits Conf.*, 2018, pp. 94–96.

[11] R. Komatsu, H. Fujii, Y. Tamura, A. Yamashita, and H. Asama, "360° depth estimation from multiple fisheye images with origami crown representation of icosahedron," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 10 092–10 099.

[12] M. Roxas and T. Oishi, "Variational fisheye stereo," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1303–1310, 2020.

[13] M. Labbé and F. Michaud, "RTAB-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *Journal of Field Robotics*, vol. 36, no. 2, pp. 416–446, 2019.

[14] S. E. Guerbas, N. Crombez, G. Caron, and E. M. Mouaddib, "Direct 3d model-based tracking in omnidirectional images robust to large inter-frame motion," in *Int. Conf. on Advanced Robotics*, 2021, pp. 505–510.

[15] K. Bouyarmane, K. Chappellet, J. Vaillant, and A. Kheddar, "Quadratic programming for multirobot and task-space force control," *IEEE Transactions on Robotics*, vol. 35, no. 1, pp. 64–77, 2019.

[16] A. Bolotnikova, P. Gergondet, A. Tanguy, S. Courtois, and A. Kheddar, "Task-space control interface for SoftBank humanoid robots and its human-robot interaction applications," in *IEEE/SICE International Symposium on System Integration*, 2021, pp. 560–565.

[17] E. Farnioli, M. Gabiccini, and A. Bicchi, "Optimal contact force distribution for compliant humanoid robots in whole-body loco-manipulation tasks," in *IEEE Int. Conf. on Robotics and Automation*, 2015, pp. 5675–5681.

[18] J. Scholz, S. Chitta, B. Marthi, and M. Likhachev, "Cart pushing with a mobile manipulation system: Towards navigation with moveable objects," in *IEEE Int. Conf. on Robotics and Automation*, 2011, pp. 6115–6120.

[19] S. Nozawa, T. Maki, M. Kojima, S. Kanzaki, K. Okada, and M. Inaba, "Wheelchair support by a humanoid through integrating environment recognition, whole-body control and human-interface behind the user," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 1558 – 1563.

[20] P. Ferrari, M. Cognetti, and G. Oriolo, "Humanoid whole-body planning for loco-manipulation tasks," in *IEEE International Conference on Robotics and Automation*, 2017, pp. 4741–4746.

[21] A. Settimi, D. Caporale, P. Kryczka, M. Ferrati, and L. Pallottino, "Motion primitive based random planning for loco-manipulation tasks," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2016, pp. 1059–1066.

[22] J. C. Vaz, N. Torres-Reyes, and P. Y. Oh, "Humanoid interaction with material-moving carts and wheelbarrows," in *IEEE International Symposium on Safety, Security, and Rescue Robotics*, 2021, pp. 21–26.

[23] I. Kumagai, M. Morisawa, T. Sakaguchi, S. Nakaoka, K. Kaneko, H. Kaminaga, S. Kajita, M. Benallegue, R. Cisneros, and F. Kanehiro, "Toward industrialization of humanoid robots: Autonomous plasterboard installation to improve safety and efficiency," *IEEE Robotics Automation Magazine*, vol. 26, no. 4, pp. 20–29, 2019.

[24] M. Murooka, I. Kumagai, M. Morisawa, F. Kanehiro, and A. Kheddar, "Humanoid loco-manipulation planning based on graph search and reachability maps," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1840–1847, 2021.

[25] M. Murooka, K. Chappellet, A. Tanguy, M. Benallegue, I. Kumagai, M. Morisawa, F. Kanehiro, and A. Kheddar, "Humanoid loco-manipulations pattern generation and stabilization control," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5597–5604, 2021.

[26] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, "Simultaneous localization and mapping: A survey of current trends in autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 2, no. 3, pp. 194–220, 2017.

[27] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual SLAM algorithms: A survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, pp. 2–11, 2017.

[28] J. Civera and S. H. Lee, *RGB-D Odometry and SLAM*. Cham: Springer International Publishing, 2019, pp. 117–144.

[29] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 1052–1067, 2007.

[30] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *IEEE/ACM International Symposium on Mixed and Augmented Reality*, Nov 2007, pp. 225–234.

[31] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: Semidirect visual odometry for monocular and multicamera systems," *IEEE Trans. on Robotics*, vol. 33, no. 2, pp. 249–265, 2017.

[32] H. Matsuki, L. von Stumberg, V. Usenko, J. Stückler, and D. Cremers, "Omnidirectional DSO: Direct sparse odometry with fisheye cameras," *IEEE Robotics and Autom. Letters*, vol. 3, no. 4, pp. 3693–3700, 2018.

[33] S. Sumikura, M. Shibuya, and K. Sakurada, "OpenVSLAM: A versatile visual SLAM framework," in *ACM International Conference on Multimedia*, 2019, pp. 2292–2295.

[34] C. Kerl, J. Sturm, and D. Cremers, "Dense visual SLAM for RGB-D cameras." in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 2100–2106.

[35] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "Elasticfusion: Real-time dense slam and light source estimation," *Int. J. of Robotics Research*, vol. 35, pp. 1697–1716, 2016.

[36] S. Wang, R. Clark, H. Wen, and N. Trigoni, "DeepVO: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *IEEE International Conference on Robotics and Automation*, May 2017, pp. 2043–2050.

[37] G. L. Oliveira, N. Radwan, W. Burgard, and T. Brox, "Topometric localization with deep learning," *Robotics Research*, pp. 505–520, 2020.

[38] J. Zubizarreta, I. Aguinaga, and J. M. M. Montiel, "Direct sparse mapping," *IEEE Trans. Rob.*, vol. 36, no. 4, pp. 1363–1370, Aug. 2020.

[39] T. Schöps, T. Sattler, and M. Pollefeys, "BAD SLAM: Bundle Adjusted Direct RGB-D SLAM," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 134–144.

[40] S. Pfeiffer, D. Ebrahimian, S. Herse, T. N. Le, S. Leong, B. Lu, K. Powell, S. A. Raza, T. Sang, I. Sawant, M. Tonkin, C. Vinaviles, T. D. Vu, Q. Yang, R. Billingsley, J. Clark, B. Johnston, S. Madhisetty,

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2024, Yokohama, Japan.

N. McLaren, P. Peppas, J. Vitale, and M.-A. Williams, "UTS unleashed! RoboCup@Home SSPL champions 2019," in *RoboCup 2019: Robot World Cup XXIII*, 2019, pp. 603–615.

[41] M. Tsuru, A. Escande, A. Tanguy, K. Chappellet, and K. Harada, "Online object searching by a humanoid robot in an unknown environment," *IEEE Robotics and Autom. Letters*, vol. 6, no. 2, pp. 2862–2869, 2021.

[42] E. Marchand and F. Chaumette, "Fitting 3D models on central catadioptric images," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 52–57.

[43] G. Caron, E. M. Mouaddib, and E. Marchand, "3D model based tracking for omnidirectional vision: A new spherical approach," *Robotics and Autonomous Systems*, vol. 60, no. 8, pp. 1056 – 1068, 2012.

[44] N. Crombez, G. Caron, and E. M. Mouaddib, "Using dense point clouds as environment model for visual localization of mobile robot," in *Int. Conf. on Ubiquitous Robots and Ambient Intelligence*, 2015, pp. 40–45.

[45] B. Wen, C. Mitash, B. Ren, and K. E. Bekris, "se(3)-TrackNet: Data-driven 6D pose tracking by calibrating image residuals in synthetic domains," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 10 367–10 373.

[46] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," in *Robotics: Science and Systems*, 2018.

[47] M. Garon and J.-F. Lalonde, "Deep 6-dof tracking," *IEEE Trans. on Visualization and Computer Graphics*, vol. 23, no. 11, pp. 2410–2418, Nov 2017.

[48] M. Wüthrich, P. Pastor, M. Kalakrishnan, J. Bohg, and S. Schaal, "Probabilistic object tracking using a range camera," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2013, pp. 3195–3202.

[49] J. Issac, M. Wuthrich, C. G. Cifuentes, J. Bohg, S. Trimpe, and S. Schaal, "Depth-based object tracking using a robust gaussian filter," *IEEE Int. Conf. on Robotics and Automation*, pp. 608–615, May 2016.

[50] K. Bouyarmane and A. Kheddar, "On weight-prioritized multitask control of humanoid robots," *IEEE Transactions on Automatic Control*, vol. 63, no. 6, pp. 1632–1647, June 2018.

[51] É. Marchand and F. Chaumette, "Virtual visual servoing: A framework for real-time augmented reality," *Computer Graphics Forum*, vol. 21, no. 3, pp. 289–298, 2002.

[52] D. Brown, "Decentering distortion of lenses," *Photogrammetric Engineering*, vol. 32, pp. 444–462, 1966.

[53] G. Blais and M. D. Levine, "Registering multiview range data to create 3d computer objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 820–824, 1995.

[54] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in *IEEE Int. Symp. on Mixed and Augmented Reality*, October 2011, pp. 127–136.

[55] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *IEEE International Conference on Robotics and Automation*, 1991, pp. 2724–2729 vol.3.

[56] A. Tanguy, A. Kheddar, and A. I. Comport, "Online eye-robot self-calibration," in *IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, 2018, pp. 68–73.

[57] K. Kaneko, H. Kaminaga, T. Sakaguchi, S. Kajita, M. Morisawa, I. Kumagai, and F. Kanehiro, "Humanoid robot HRP-5P: An electrically actuated humanoid robot with high-power and wide-range joints," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1431–1438, 2019.

[58] Y. Wang, N. Dehio, A. Tanguy, and A. Kheddar, "Impact-aware task-space quadratic-programming control," *arXiv preprint arXiv:2006.01987*, 2020.

Kevin Chappellet received the MS in computer sciences and applied mathematics from Joseph Fourier University, Grenoble, France, in 2014 and the PhD in robotics from the University of Montpellier, Montpellier, France, in December 2021. He completed his PhD program in the Joint Robotics Laboratory, National Institute of Advanced Industrial Science and Technology, Tsukuba, Japan. From November 2014 to March 2019, he was a Research Engineer with the Interactive Digital Human Group, CNRS-UM Laboratory of Informatics, Robotics, and Microelectronics, Montpellier, France, involved in numerous European projects such as VERE, Robohow and Comanoid with HRP-4 humanoid robot experiments. His focus is on autonomous loco-manipulation based on visual feedback.

Masaki Murooka received the BE, MS, and PhD degree in information science and technology from The University of Tokyo, Japan, in 2013, 2015, and 2018, respectively. He was a project assistant professor in the Department of Mechano-Informatics at The University of Tokyo from 2018 to 2020. He joined the CNRS-AIST Joint Robotics Laboratory in the National Institute of Advanced Industrial Science and Technology (AIST) in 2020 as a researcher. His research interest includes the motion planning and control of humanoid robots.

Guillaume Caron received the BS in Computer Science in 2005 from the Universite de Picardie Jules Verne (UPJV), Amiens, France, the MS in Computer Science in 2007 from the Universite de Reims Champagne-Ardennes, Reims, France and the PhD in robotics and the Habilitation degree from UPJV in 2010 and 2019. He has been Associate Professor since 2011 at UPJV and CNRS delegate since 2019 at the CNRS-AIST Joint Robotics Laboratory (JRL), IRL, Tsukuba, Japan. He has been the deputy director of JRL since 2022 after being the leader of the Robotic Perception group of the MIS laboratory (UPJV) from 2016 to 2020. Before, he was a postdoc fellow at Inria, Rennes, France (2010-2011) and stayed at the University of Osaka, Japan (Apr.-May 2013). His research interests include artificial vision for robotics, real-time visual tracking and servoing, new vision sensors and digital heritage. He was the Chair of the Technical Committee 19 "Computer vision for cultural heritage applications" of the International Association for Pattern Recognition (IAPR) from 2018 to 2022, year when he became an elected member of the AFRIF's (French IAPR) board of directors and an IEEE Senior Member.

Fumio Kanehiro received the BE, ME, and PhD in engineering from The University of Tokyo, Japan, in 1994, 1996, and 1999, respectively. He was a Research Fellow of the Japan Society for the Promotion of Science in 1998-1999. In 2000, he joined the Electrotechnical Laboratory, Agency of Industrial Science and Technology, Ministry of Industrial Science and Technology (AIST-MITI), later reorganized as National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Japan. From April 2007, he was a visiting researcher at the LAAS-CNRS for one year and three months. He is currently Director of CNRS-AIST JRL (Joint Robotics Laboratory), IRL, AIST. His research interests include the software frameworks and whole body motion planning of humanoid robots.

Abderrahmane Kheddar (F'22) received the BS in Computer Science degree from the Institut National d'Informatique (ESI), Algiers, the MSc and PhD degree in robotics, both from Sorbonne University, Paris. He is presently Directeur de Recherche at CNRS, within the CNRS-AIST Joint Robotics Laboratory (JRL), IRL, Tsukuba, Japan, and at Interactive Digital Humans (IDH) team at CNRS-University of Montpellier LIRMM, France. His research interests include haptics, humanoids and thought-based control using brain machine interfaces. He is a founding member of the IEEE/RAS chapter on haptics, the co-chair and founding member of the IEEE/RAS Technical committee on model-based optimization, he is a member of the steering committee of the IEEE Brain Initiative, Editor of the IEEE Robotics and Automation Letters, Founding member and Deputy Editor-in-Chief of Cyborg and Bionics System. He was Editor of the IEEE Transactions on Robotics (2013-2018) and within the editorial board of other robotics journals; he is a founding member of the IEEE Transactions on Haptics and served in its editorial board during three years (2007-2010). He is an IEEE Fellow, an AAIA Fellow, member of the National Academy of Technology of France and knight of the national order of merits of France.