Performing manufacturing tasks with a mobile manipulator: from motion planning to sensor based motion control

Joseph Mirabel, Florent Lamiraux, Thuc Long Ha, Alexis Nicolin, Olivier Stasse, Sébastien Boria

Performing manufacturing tasks with a mobile manipulator: from motion planning to sensor based motion control.

Joseph Mirabel[1], Florent Lamiraux[1], Thuc Long Ha[1], Alexis Nicolin[1,2], Olivier Stasse[1] and Sébastien Boria[2]

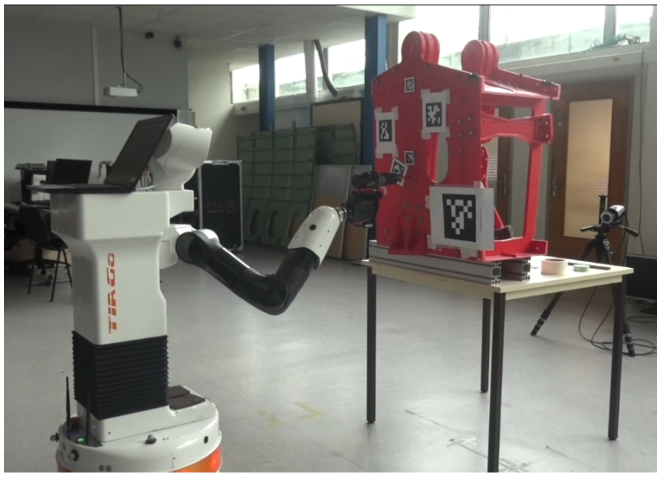


Fig. 1. Tiago robot performing a deburring task on an aircraft motor pylon mockup using visual servoing. The robot grasps a standard driller using its multi-purpose hand.

***Abstract*— We present a framework combining mobile manipulation planning and sensor based motion control. The result of manipulation planning is a reference robot trajectory composed of segments. For each segment, a hierarchical task based controller is automatically built. Some of the segments correspond to motions where some parts of the robot and of the environment are known to be close to each other. The corresponding controller implements a task of relative pose between these elements. The task error is measured by vision sensors using AprilTags. The higher priority of this task with respect to the reference joint trajectory greatly improves the accuracy of the task affected by the poor accuracy of the robot base localization and other sources of error.**

I. INTRODUCTION

Most automated manufacturing tasks in industry today are performed by very accurate manipulator arms with fixed base or computer numerical control (CNC) machining. High accuracy in the task execution is obtained by the repeatability of the machines. This way of production is known to require heavy robots and high cost infrastructure including civil engineering to be deployed. Moreover, most of these systems are programmed by hand by human operators.

Some research work today investigates the possibility to perform simple manufacturing tasks using mobile manipulators in factory floors populated by human operators [19], [6]. Those solutions would provide several benefits, among which:

- no need for costly infrastructures to fix heavy robots on the ground,
- no need for specific tools designed to be rigidly fixed at a robot end-effector.

One of the main challenges in this approach is that the poor base localization of the mobile manipulator does not enable the robot to perform in open loop any task requiring more than 10 cm accuracy. The manufacturing task thus requires some visual servoing to be performed with a reasonable accuracy.

In this paper, we report on recent work that we have done in this purpose. We illustrate this work on two different tasks performed by two different robots. The first task is a deburring task where a mobile manipulator is requested to visit some holes of a part with a deburring tool mounted on a standard drill. The second task consists of a humanoid robot manipulating a box to grasp it and put it upside down on a table.

The main contribution of the paper is a methodology that combines manipulation planning in the configuration space of the robot and objects and visual servoing based controllers at the execution level. The method automatically maps a controller to each segment of the initially planned trajectory. Along trajectory segments where parts of the robot and of the environment are known to be close to each other, the controller implements a task of relative pose between those parts.

A. *Manipulation planning*

Manipulation planning is a class of path planning problems where some robots manipulate some objects with grippers, in an environment cluttered with obstacles. Manipulation constraints raised by the fact that objects cannot move if they are not grasped and that they are rigidly attached to a gripper when they are grasped define several foliations of the configuration space of the whole system (robots + objects). From a given configuration with a given set of grasps, the system can only move on a sub-manifold of the configuration space called a leaf.

Since the pioneering paper [23], manipulation planning has been tackled using random sampling based method [9], [13] in order to explore the leaves of those foliations [20]. Manipulation planning is traditionally decomposed into several subdomains: Rearrangement planning [11], [18], [14] where the goal specifies only the final position of the objects,

*This work has been supported by Airbus SAS in the framework of the common laboratory Rof4Fam

[1]LAAS-CNRS University of Toulouse, 31077 Toulouse, France firstname.lastname@laas.fr

[2]Airbus Defense and Space, Toulouse, France

multi-arm motion planning [7], [8], [5], and navigation among movable obstacles.

B. Sensor-based manipulation planning and control

[24] addresses the problem of rearrangement planning for a disc robot manipulating circular objects in the plane. The paper presents impressive experimental results with a quadruped robot equipped with a lidar and avoiding unexpected obstacles. [21] proposes a method to plan a sequence of controllers that move a system composed of two robots and one object from an initial state to a goal state in a dynamic environment. The algorithm performs a random exploration of the configuration space, using simulated controllers as a steering method. The controllers are implemented using eTaSL/eTC framework [1]. Our work share similarities with this latter paper. The main differences are that we do not explicitly address dynamic environments, but we implement visual servoing.

C. Visual servoing

[3], [4] provide an overview of the state of the art in this domain until 2007. The second reference mentions the case of moving target only at the very end. In a more recent work [2], the authors perform a grasp task on a Romeo humanoid robot, tracking the position of the hand and of the object to grasp with a camera. In this latter work, there are no obstacles and collision of the hand with the object during the approach phase is not addressed. [10] plans a kinodynamic motion for the camera frame to bind an initial and a final configurations using an RRT algorithm. For each extension of the tree with an elementary camera motion, they compute the motion of the robot by inverse kinematics and check this motion for collision.

D. Contribution

In this paper, we propose a novel approach to manipulation planning and control for a robot with a high number of degrees of freedom in the presence of obstacles. The approach is broken down into three steps. In the first step, a manipulation planning algorithm computes a manipulation planning path starting from the initial configuration and reaching a goal configuration or set of configurations. In the second step, the manipulation path is decomposed into segments. To each segment a controller taking into account both the planned path and visual features attached to objects or robot bodies is associated. These controllers regulate sequences of tasks with priority orders [22]. In the third step, a finite state machine keeps track of the current segment and selects the controller accordingly. Note that a preliminary version of this work has been described in [17]. However, in this previous work, no visual servoing was implemented.
The paper is organized as follows. In Section II we provide a brief description of the manipulation planning framework that we have developed for the past few years, stressing on features that are of interest for the paper. In section III, we describe our sensor based motion control framework based on hierarchical task control. In Section IV, we describe our experimental setup and provide some experimental data and analyses. Section **??** concludes the paper.

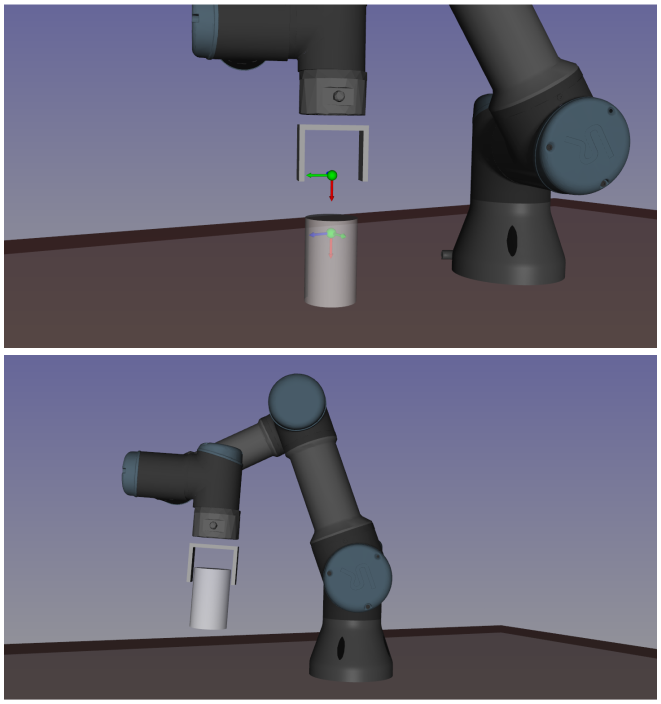


Fig. 2. Top: a frame attached to the robot gripper is called a *gripper*. A frame attached to an object is called a *handle*. Bottom: a pair (gripper, handle) defines a sub manifold of configurations called a *grasp*.

II. MANIPULATION PLANNING

In this section, we briefly describe our manipulation planning framework. The role of this framework is to automatically compute collision-free feasible trajectories for the system that satisfies the task to perform. The task may consist in moving objects from an initial pose to a final pose or to move a tool to some specified poses in order to perform a manufacturing task (drilling, deburring,...). This part is a prerequisite for the contribution of this paper. For this reason, we provide only a brief description. See [12] for a complete description of the framework.

We consider a robot with configuration space denoted as $\mathcal{C}_r$ and $n_o \geq 1$ objects of configuration spaces $SE(3)$[1]. The configuration space of the whole system is the Cartesian product of the configuration spaces of the robot and objects: $\mathcal{C} = \mathcal{C}_r \times SE(3)^{n_o}$. Some static obstacles are present in the workspace of the robot.

A. Gripper, handle, grasp

We define frames attached to robot links or to objects that we call *gripper* or *handle*. A grasp corresponds to the association of a gripper with a handle. Each grasp (or combination of grasps) defines a sub-manifold in $\mathcal{C}$ containing the configurations where the *gripper* and *handle* frames coincide. Figure 2 gives an example.

[1]$SE(3)$ is the group of rigid-body transformations

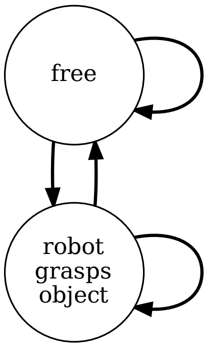


Fig. 3. Constraint graph defined by a robot with one gripper manipulating an object with one handle, as represented in Figure 2. Node ”free” corresponds to configurations where the robot does not hold the object. Node ”robot grasps object” corresponds to configurations where the robot holds the object. Edges link states that have non empty intersections. Figure 4 middle displays a configuration that belongs to the intersection of both states.

B. *Constraint graph*

We represent a manipulation problem by a graph called *constraint graph*. The nodes of the graph are defined by all the possible combinations of grasps as defined above. Edges link nodes that share configurations. Figure 3 illustrates the notion of constraint graph on a simple example.

C. *Pre-grasp*

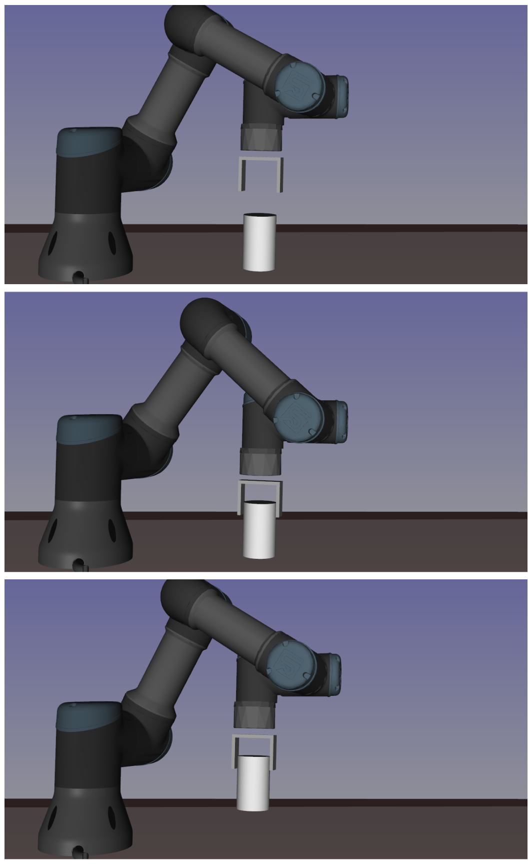
From a motion planning point of view, computing a collision-free motion to grasp an object is a difficult step since the final configuration is very close to the configuration space obstacle. Only a small portion of approaching directions of the gripper towards the object yield trajectories where the gripper does not collide the object. Hence, grasping and lifting an object requires to go through what is called a *narrow passage* in the configuration space. Narrow passage are known to raise difficult issues for random path planning methods like RRT or PRM.

To overcome this issue, we force the approaching direction of the gripper towards the object by adding way-point states in the constraint graph as explained in Figure 4. An example of a path planned by our manipulation planning framework can be seen by following this link.

D. *Result of manipulation planning*

We do not provide more details about the algorithm that solves manipulation planning. We refer to [12] for details. The result of manipulation planning is a trajectory parameterized by time in the configuration space of the system robot, objects. The trajectory is partitioned into parts each of which corresponds to an edge of the constraint graph as illustrated by Figure 5.

In the next section, we explain how the edge associated to each part of the trajectory defines a specific controller.



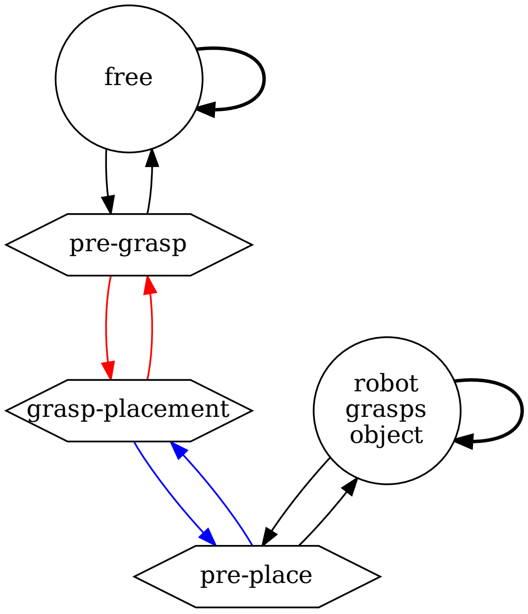


Fig. 4. Constraint graph with way-points. Way-point states are added to the initial constraint graph to force the approaching direction of the gripper and the lifting direction of the object. The pictures respectively represent way-point states called *pre-grasp*, *grasp-placement* and *pre-placement* by hexagonal boxes.

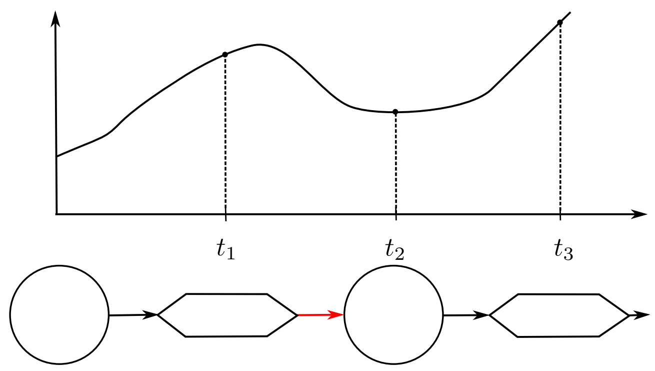


Fig. 5. The result of the manipulation planning algorithm is a trajectory in the configuration space of the system. The trajectory is partitioned into parts each of which corresponds to an edge of the constraint graph.

III. SENSOR BASED MOTION CONTROL

The rationale of this section is the following. Let us be given the result of a manipulation planning problem as a sequence of parts each of which belongs to an edge of the constraint graph, as in Figure 5. Along parts of the trajectory corresponding to

1) edges between *pre-grasp* and *grasp-placement* (in red in Figure 4), the gripper moves close to the object to be grasped,
2) edges between *grasp-placement* and *pre-place* (in blue in Figure 4), the object moves close to the contact surface it was lying on.

Executing the whole trajectory in open loop with a mobile manipulator would almost surely fail because of a collision between the gripper and the object or between the object and the contact surface. The main idea of this paper is to selectively control the relative pose

1) between the gripper and the object to be grasped along red edges,
2) between the object and the contact surface along blue edges.

To this aim, we use a *task based hierarchical controller* as described in [15] that implements the algorithm described in [22].

A. Hierarchical task based controller

The controller is based on the following concepts:

- *task:* a mapping from $\mathcal{C}$ to $\mathbb{T}$ a vector space or a Lie-group (usually $SE(3)$). It can represent the pose of an end effector, the posture of the robot, the pose of an object;
- *task reference:* a mapping from $\mathbb{R}$ to $\mathbb{T}$ that represents the desired trajectory of the task along time.
- *error:* a mapping from $\mathbb{T}\times\mathbb{T}$ to a vector space that maps to zero pairs of identical task values. For instance, if the task is the pose ($\mathbb{T} = SE(3)$) of a robot end effector, and if T_1, T_2 are two elements of $SE(3)$,

$$\mathbf{e}_{SE(3)}(T_1, T_2) = \log\left(T_1^{-1}T_2\right) \tag{1}$$

is the screw velocity $(v, \omega) \in \mathbb{R}^6$ that moves T_1 to T_2 in unit time. Particularly, the error is equal to zero iff $T_1 = T_2$. If $\mathbb{T} = \mathbb{R}^n$ for a given positive integer n,

$$\mathbf{e}_{\mathbb{R}^n}(T_1, T_2) = T_2 - T_1. \tag{2}$$

If we consider a planned trajectory Γ as a mapping from $\mathbb{R}$ to $\mathcal{C}$, and a task T with output space $\mathbb{T}$, we define the *task reference associated to* T for the trajectory Γ as the mapping from $\mathbb{R}$ to $\mathbb{T}$:

$$T^*(t) \triangleq T(\Gamma(t)) \tag{3}$$

and the *error for this task along* Γ as the mapping from $\mathcal{C}\times\mathbb{R}$ to $\mathbb{R}^n$ for a given n:

$$\mathbf{e}(\mathbf{q}, t) \triangleq \mathbf{e}_{\mathbb{T}}(T(\mathbf{q}), T^*(t)). \tag{4}$$

The error may be seen as the *difference* between the value of the task (as measured by sensors) and the desired value provided by the reference trajectory.

a) The controller: given an ordered list of tasks, and a measurement by sensors for each task, the controller computes a control variable (here, the joint velocity vector) in such a way that as many task errors as possible converge to 0:

1) if there exists a control that makes all error decrease, the solver will compute such a control,
2) if some tasks are not compatible (making one decrease implies that the other one increases), the controller will make the highest priority task error decrease.

The controller computes a control at the controller frequency. See [22] for a description of the algorithm.

In our case, as described below, following the reference trajectory computed by the manipulation planner is a low priority task (*posture task*), while in some controllers, the relative pose of the end effector with respect to an object (measured by vision sensors) is a higher priority task. Due to modeling error (poor calibration of the kinematic chain, inaccuracy of sensors), these tasks are incompatible since they both specify the position of the end effector with different references. As the latter has a higher priority, the controller will track at best the relative position of the end effector while remaining as close as possible to the planned trajectory.

B. Synthesis of the controllers

As mentioned above, we associate a controller to each part of the trajectory computed by the manipulation planning algorithm. The controller depends on the edge of the constraint graph the part is associated to.

Each controller contains at most 3 tasks in decreasing order of priority as follows:

a) System specific task: Possible robot specific constraints are inserted at the highest priority level (level 1). This task can be for instance quasi-static equilibrium of a legged humanoid robot, or a predefined trajectory of the base of a mobile manipulator.

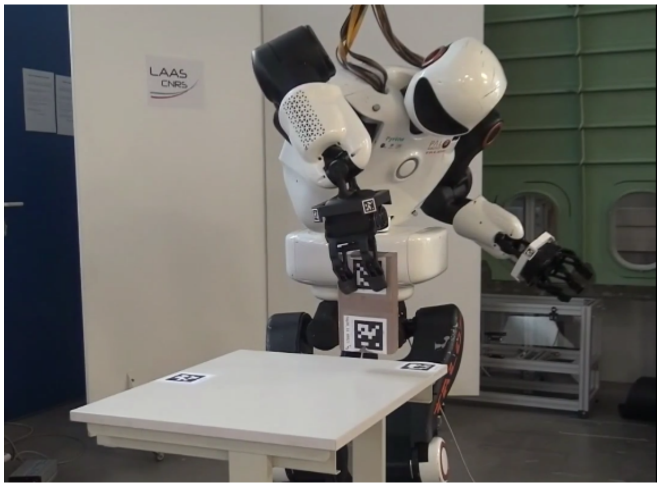


Fig. 6. Example of the effect of visual servoing on the motion execution accuracy. Eventhough the box has slipped in the gripper, the controller successfully positions the box vertically before putting it on the table.

b) Edge specific task: they are defined as follows.

- For general edges (black in Figure 4), there is no specific task.
- For edges linking a *pre-grasp* to a *grasp-placement* state (red in Figure 4), the *edge specific* task is a task of relative pose of the gripper with respect to the object to be grasped.
- For edges linking a *grasp-placement* to a *pre-place* state, the *edge specific* task is a task of relative pose of the object with respect to the contact surface.

c) Posture task: The posture task takes values in $\mathbb{R}^n$ where n is the number of actuated degrees of freedom of the robot. The value of the task for a given configuration $\mathbf{q}$ is obtained by extracting from $\mathbf{q}$ the parameters corresponding to the actuated degrees of freedom.

Note that for each task, the reference is the value provided by the planned trajectory.

IV. EXPERIMENTAL RESULTS

We have applied our method to two different robots performing different tasks. The first one is the humanoid robot Pyrene manipulating a box lying on a table (Figure 6). The second one is a Tiago robot performing deburring tasks on a aircraft part (Figure 1). The detection of AprilTags is performed by ViSP software platform [16].

A. Pyrene manipulates a box

The experimental setup is composed of the humanoid robot, a table and a wooden box both equipped with April-Tags [25].

In this section, we show the execution of a manipulation sequence where the robot is asked to flip a box upside down on a table. The reference path is planned and optimized by our software platform `HPP` as explained in section II.

AprilTags are stuck to the robot grippers, to the object and to the table. As explained in the previous section, all controllers take as highest priority task the quasi-static equilibrium task of the robot (position of the feet and of the center of mass).

- between pre-grasp and grasp-placement, the relative pose of the gripper with respect to the box is inserted in second priority order in the controller,
- between pre-place and grasp-placement, the relative position of the box with respect to the table is inserted in second priority order in the controller.

This movie shows the execution of the motion on the real robot. Figure 7 displays plots of the error of the relative pose between the robot gripper and the box.

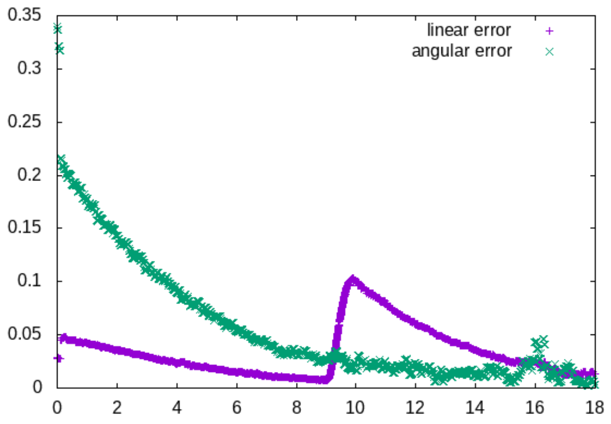


Fig. 7. Plots of the linear and angular errors of the visual servoing task along the approaching box motion. The angular error decreases exponentially. The linear error decreases, then increases between 9 and 10s and then decreases exponentially again. The increase is due to the descending motion of the gripper to grasp the box. The gain of the visual servoing task is low because of the small frequency of the visual servoing process. This implies a delay in following the descending trajectory.

B. Tiago performs deburring tasks

The experimental setup is composed of a mobile manipulator Tiago, a drill and a 3D printed mockup of an aircraft part. The robot is asked to insert a deburring tool fixed to the drill inside predefined holes of the aircraft part.

The robot grasps the drill via a *gripper* frame attached to the hand of the robot and a *handle* frame attached to the drill. Note that the drill is placed manually in the hand of the robot. The relative position of the drill with respect to the hand is therefore not accurately known.

Each deburring task is represented by a *handle* and a *gripper* is attached to the deburring tool inserted in the drill. The deburring task is thus performed by following trajectories between pre-grasp and grasp for these virtual grasps. See Figure 8 for an illustration.

The output of the manipulation planning algorithm is a trajectory composed of

1) base motions with the arm folded in a default configuration,
2) arm motions from the folded configuration to deburring *pre-grasp* states,
3) back and forth straight motions of the tool inside the holes

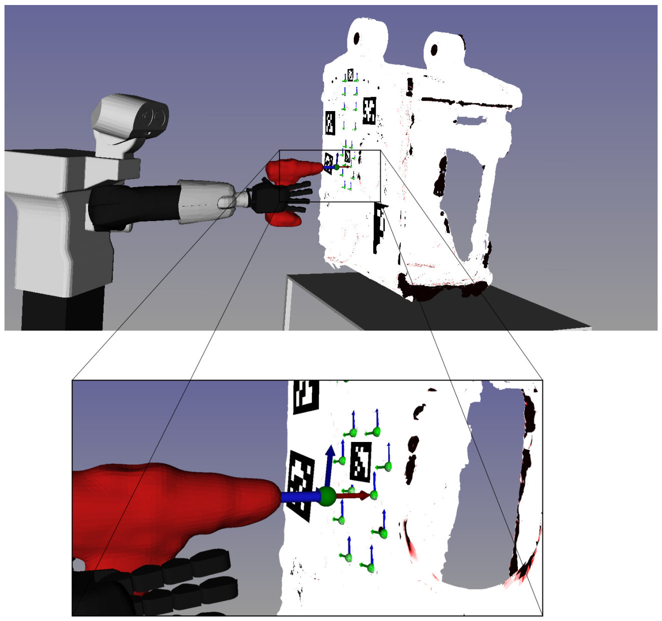


Fig. 8. Deburring tasks are represented by virtual grasp – pre-grasp pairs. A *gripper* frame is attached to the deburring tool and *handle* frames are attached to each hole of the part. Part and drill models have been built using an RGBD sensor.

4) arm motion to the folded configuration.

As explained in Section III-B, motions of type 2 are controlled by a visual servoing task of the drill pose with respect to the part. To this aim, the drill and part are equipped with AprilTags.

The video attached to the paper illustrates the various steps of the method.

C. *Software*

The software implementing the two demonstrations is composed of

1) Humanoid Path Planner for the manipulation planning part,
2) Agimus for mapping controllers to edges of the constraint graph and for implementing the finite-state machine controlling the whole demo,
3) the Stack of Tasks for implementing real-time hierarchial based controllers on-board the robots.

All these software projects are open-source.

REFERENCES

[1] E. Aertbeliën and J. De Schutter, "Etasl/etc: A constraint-based task specification language and robot controller using expression graphs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 09 2014.

[2] D. J. Agravante, G. Claudio, F. Spindler, and F. Chaumette, "Visual servoing in an optimization framework for the whole-body control of humanoid robots," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 608–615, Apr. 2017. [Online]. Available: https://hal.inria.fr/hal-01421734

[3] F. Chaumette and S. Hutchinson, "Visual servo control, part i: Basic approaches," *IEEE Robotics and Automation Magazine*, vol. 13, no. 4, pp. 82–90, December 2006.

[4] ——, "Visual servo control, part ii: Advanced approaches," *IEEE Robotics and Automation Magazine*, vol. 14, no. 1, pp. 109–118, March 2007.

[5] A. Dobson and K. Bekris, "Planning representations and algorithms for prehensile multi-arm manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015.

[6] A. Dömel, S. Kriegel, M. Kaßecker, M. Brucker, T. Bodenmüller, and M. Suppa, "Toward fully autonomous mobile manipulation for industrial environments," *International Journal of Advanced Robotic Systems*, vol. 14, no. 4, p. 1729881417718588, 2017.

[7] M. Gharbi, J. Cortés, , and T. Siméon, "Roadmap composition for multi-arm systems path planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Saint-Louis, USA, 2009.

[8] K. Harada, T. Tsuji, and J.-P. Laumond, "A manipulation motion planner for dual-arm industrial manipulators. in proceedings of," in *IEEE International Conference on Robotics and Automation*, Hongkong, China, 2014, pp. 928—934.

[9] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, August 1996.

[10] M. Kazemi, K. K. Gupta, and M. Mehrandezh, "Randomized kinodynamic planning for robust visual servoing," *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1197–1211, Oct 2013.

[11] A. Krontiris and K. Bekris, "Dealing with difficult instances of object rearrangement," in *Robotics Science and Systems*, Roma, Italy, 2015.

[12] F. Lamiraux and J. Mirabel, "Prehensile Manipulation Planning: Modelling, Algorithms and Implementation," Nov. 2020. [Online]. Available: https://hal.laas.fr/hal-02995125

[13] S. M. Lavalle and J. J. Kuffner, "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, May 2001.

[14] P. Lertkultanon and Q.-C. Pham, "A single-query manipulation planner," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 198–205, 2015.

[15] N. Mansard, O. Stasse, P. Evrard, and A. Kheddar, "A versatile generalized inverted kinematics implementation for collaborative working humanoid robots: The stack of tasks," in *International Conference on Advanced Robotics (ICAR)*, 2009.

[16] E. Marchand, F. Spindler, and F. Chaumette, "Visp for visual servoing: a generic software platform with a wide class of robot control skills," *IEEE Robotics and Automation Magazine*, vol. 12, no. 4, pp. 40–52, December 2005.

[17] A. Nicolin, J. Mirabel, S. Boria, O. Stasse, and F. Lamiraux, "Agimus: a new framework for mapping manipulation motion plans to sequences of hierarchical task-based controllers," in *IEEE/SICE International Symposium on System Integration*, Honolulu, United States, Jan. 2020. [Online]. Available: https://hal.laas.fr/hal-02466543

[18] J. Ota, "Rearrangement of multiple movable objects-integration of global and local planning methodology," vol. 2, 2004, pp. 1962–1967.

[19] M. A. Roa, D. Berenson, and W. Huang, "Mobile manipulation: Toward smart manufacturing [tc spotlight]," *IEEE Robotics Automation Magazine*, vol. 22, no. 4, pp. 14–15, 2015.

[20] P. S. Schmitt, W. Neubauer, W. Feiten, K. M. Wurm, G. V. Wichert, and W. Burgard, "Optimal, sampling-based manipulation planning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3426–3432. [Online]. Available: http://ais.informatik.uni-freiburg.de/publications/papers/schmitt17icra.pdf

[21] P. S. Schmitt, F. Wirnshofer, K. M. Wurm, G. V. Wichert, and W. Burgard, "Modeling and planning manipulation in dynamic environments," in *2019 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2019. [Online]. Available: http://ais.informatik.uni-freiburg.de/publications/papers/schmitt19icra.pdf

[22] B. Siciliano and J.-J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *International Conferenceon Advanced Robotics*, 1991, pp. 1211–1216.

[23] T. Siméon, J.-P. Laumond, J. Cortés, and A. Sahbani, "Manipulation planning with probabilistic roadmaps," *International Journal of Robotics Research*, vol. 23, no. 7/8, July 2004.

[24] V. Vasilopoulos, T. Topping, W. Vega-Brown, N. Roy, and D. Koditschek, "Sensor-based reactive execution of symbolic rearrangement plans by a legged mobile manipulator," in *2018 IEEE/RSJ International Conference on Intelligent Robotic Systems (IROS)*, 10 2018.

[25] J. Wang and E. Olson, "AprilTag 2: Efficient and robust fiducial detection," 2016.