A Generic Optimization Based Cartesian Controller for Robotic Mobile Manipulation

Emilia Brzozowska[1], Oscar Lima[2], Rodrigo Ventura[3]

***Abstract*— Typically, the problem of robotic manipulation is divided among two sequential phases: a planning one and an execution one. However, since the second one is executed in open loop, the robot is unable to react in real time to changes in the task (e.g. moving object). This paper addresses the mobile manipulation problem from a real-time, closed loop perspective. In particular, we propose a generic optimization-based Cartesian controller, that given a continuous monitoring of the goal, determines the best motion commands. We target our controller to a robotic system comprising an arm and a mobile platform. However, the approach can in principle be extended to more complex mechanisms. The approach is based on shifting the problem to velocity space, where end effector velocity is a linear function of joint and base platform velocities. Our approach was quantitatively evaluated both on simulation and on a real service robot. It was also integrated into a mobile service robot architecture targeting domestic tasks and evaluated on the RoboCup@Home scientific competition. Our results show that the controller is able to reach random arm configurations with a high probability of success.**

I. Introduction

A. Motivation

Object grasping in robots is typically approached by using separate offline path planning and open loop execution methods, which expose some disadvantages. For instance, during trajectory execution, the robot is not sensitive to changes in the task. For instance, if the target object pose is changed, the robot should ideally deal successfully with those situations by adjusting accordingly. Moreover most off-the-shelf methods typically control only the arm and do not take into account the robot base. These limitations motivated finding a simple but effective approach to both control the joint arm and base robot system in real-time, in close loop with the robot perception pipeline.

B. Problem Statement

In this paper we consider a robotic arm mounted on top a mobile robot base. The arm may be either under-actuated (less than 6 DoF) or with redundancy (more than 6 DoF). Also, the robot base may be or not omnidirectional. The problem is defined as, given a 6D target pose (position and orientation), the determination of the best joint velocity for both the arm and the base. Even though the developed solution was integrated and tested on the MBot robot [4] (Fig. 1), it can be easily applied to different robot configurations.

Fig. 1: Robot attempting to grasp dishwasher handle.

C. Contribution

Contributions of this paper are:

- a generic real time base and arm closed loop Cartesian velocity controller that is able to achieve a target pose, by moving the robot base and the arm jointly.
- an experimental evaluation of the developed controller both in simulation and in a real robot.
- a ROS based open source implementation available under[1].

II. Related Work

Many research groups currently focus on the development of safe compliant systems for mobile manipulation in unstructured collaborative environments. A prominent example can be found in the work presented by Nimbro@Home team [7], which developed an arm controller based on a differential inverse kinematics method to follow computed trajectories. The team solution for the inverse kinematics redundancy problem uses null-space optimization of the previously implemented cost function. Optimization criterion include convenient joint angles configuration and a penalty function for getting close to the joints limits.

Nieuwenhuisen et al. [5] use the kinodynamic motion planning by interior-exterior cell exploration algorithm [9] for the motion planning. They filter grasp poses and motion

[1]Emilia Brzozowska is with Department of Automatic Control and Robotics, UST AGH, Cracow, Poland emilia.brzozowska@hotmail.com

[2]Oscar Lima is with DFKI Robotics Innovation Center, Bremen, Germany oscar.lima@dfki.de

[3]Rodrigo Ventura is with the Institute for Systems and Robotics, Instituto Superior Tecnico, Av. Rovisco Pais 1, 1049-001 Lisbon, Portugal rodrigo.ventura@isr.tecnico.ulisboa.pt

[1]https://github.com/EmiliaBrzozowska/optimisation_cartesian_controller

paths before execution against a height map, finding a collision-free solution of the inverse kinematics.

Chitta et al. [2] use randomized sampling based motion planners from the Open Motion Planning Library (OMPL) [8]. The approach for trajectory execution includes a state machine concept, that involves moving the sensors of the robot to maintain visibility of the robotic arm, which is executing the planned motion. The controller tracks and executes the desired motion simultaneously.

In the work presented by Stuckler et al. [6] mobile manipulation and motion control problem is approached separately for the robots wheeled mobile base and for the robotic arm. Control directions of individual wheel velocities are coming from inverse kinematics analytic solutions. For the arms they developed a compliance controller with the servo actuators having a limited torque. Collision free path is obtained with implemented differential inverse kinematics.

Ciocarlie et al. [3] research is another example of kinodynamic motion planning usage by interior-exterior cell exploration algorithm from the OMPL Library. In this approach, collision awareness is also provided with an inverse kinematics based method. The motion planner generates paths that are processed by a trajectory optimization planner. In this study, they pass a previously generated path from the OMPL as an initial condition for the optimization problem. The controller is responsible for eventual re-planning of the motion path.

III. Approach

Before an object can be manipulated, the robot needs to perceive the object 6D pose by using a perceptual module. The perception problem on itself, including the object detection, 6D pose estimation from (noisy) sensor data and object classification are considered out of the scope of this work. For the experimental part we use an off-the-shelf solution.

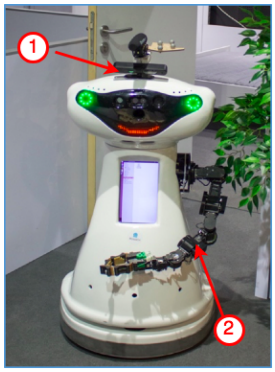


Fig. 2: MBot: 1) Head camera used in this project 2) Robot manipulator used in the experimental part of this work

In order to address the proposed challenges, we developed a solution that closes the loop between perception and the robot actuation. Our approach allows the robot to overcome problems such as changing the object goal pose in real time. Controller design details are provided in the next section.

A. Controller Design

In our proposed architecture (see Fig. 3) we consider a real-time closed loop between the perception module and the arm and base actuation. The pose error between the current and the target end effector poses is computed first. Then, the arm controller is responsible for minimizing this error by determining the feasible joint velocities to do so. We include in the set of joints both the arm joints and the mobile base actuation. Since the optimization is done in velocity space, the transformation between Cartesian space and joint space is linear, given by the Jacobian matrix of the direct kinematics transformation, in combination with the mobile base differential kinematics. The minimization problem is formulated as a constrained optimization problem, where the constraints are the joint velocity limits. Since the optimization is formulated in Cartesian space, our approach can be classified as a Position-Based Visual Servo (PBVS) control one [1].

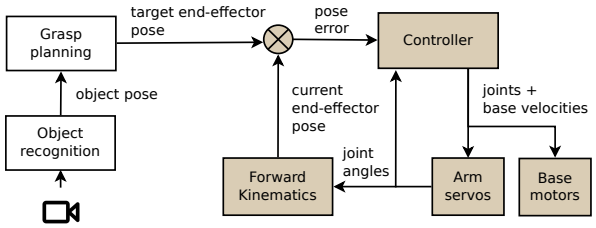


Fig. 3: System architecture showing the pipeline from perception to actuation, in closed loop with the proposed controller. The blocks covered in this paper are shaded.

Since the minimization is done in velocity space, we chose to include in the cost function the quadratic error between the Cartesian velocity resulting from the joint velocities and an ideal velocity that would achieve the target pose in a single time step. In addition to this error, we also include the following penalty terms:

1) a logarithmic barrier on the arm joint angles to avoid arm configurations close to joint limits,
2) another logarithmic barrier to avoid the robot base to be oriented opposite to the target goal, and
3) a L_1 regularizer term to tune the actuation balance among all the joints.

All of these three terms are not fundamental to the method, but contribute to a more smooth and natural behavior by the robot. The later term is particularly relevant in redundant kinematic chains, since in that case the optimization problem is ill-posed (i.e., it has multiple solutions).

B. Optimization Problem

The problem is formulated on local coordinates, that is, with respect to the kinematic frame of the base robot, hereby called *base frame*. Both the arm kinematic chain and the object target goal of the arm are assumed to be defined with respect to this frame.

The unknowns in the optimization problem are the joint velocities. In this paper we consider an omnidirectional base,

controlled in velocity mode, and an arm with N joints. The base velocity reference is considered Cartesian and expressed on the base frame, where $\dot{b}_x$ and $\dot{b}_y$ are the translational velocity reference in X and Y, and $\dot{b}_\Theta$ is the angular velocity. The arm joints are also considered to be controlled in velocity, with references $\dot{\Theta} = (\dot{\theta}_1, \ldots, \dot{\theta}_N)$ for the N joints.

Then, using the Jacobian of the arm, the end effector velocity, in translational (v_x, v_y, v_z) and angular $(\omega_x, \omega_y, \omega_z)$ terms, expressed on the base frame, is given by

$$V = \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} -\dot{b}_\Theta y \\ \dot{b}_\Theta x \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + J(\Theta)\dot{\Theta} + \begin{bmatrix} \dot{b}_x \\ \dot{b}_y \\ 0 \\ 0 \\ 0 \\ \dot{b}_\Theta \end{bmatrix} \quad (1)$$

where x and y are the projection of the end effector position on the X-Y plane, on the base reference frame, and $J(\Theta)$ is the arm Jacobian, in function of the current joint angles $\Theta = (\theta_1, \ldots, \theta_N)$.

Let P_c and R_c be the current end effector position and orientation, as a rotation matrix, and P_g and R_g be the goal position and orientation. Let us consider the velocity reference, that is, the velocity that would attain the target pose in one time step, to be defined as

$$V_r = \frac{1}{\Delta t} \begin{bmatrix} P_g - P_c \\ \log(R_g\, R_c^T) \end{bmatrix} \quad (2)$$

where Δt is the time step of the controller and $\log()$ is the matrix logarithm[2]

The proposed constrained optimization problem is the following:

$$\begin{array}{rl} \text{Minimize} & ||V_r - V||^2 + P(\dot{\Theta}) + B(\gamma) \\ & +W(\dot{\Theta}, \dot{b}_x, \dot{b}_y, \dot{b}_\Theta) \\ \text{w.r.t. :} & \dot{\Theta}, \dot{b}_x, \dot{b}_y, \dot{b}_\Theta \\ \text{subject to} & \dot{\Theta}^{Min} \leq \dot{\Theta} \leq \dot{\Theta}^{Max} \\ & \dot{b}^{Min} \leq (\dot{b}_x, \dot{b}_y, \dot{b}_\theta) \leq \dot{b}^{Max} \end{array} \quad (3)$$

where $\dot{b}^{Min}$ and $\dot{b}^{Max}$ are the bounds for base velocities and $\dot{\Theta}^{Min}$ and $\dot{\Theta}^{Min}$ are joint velocity bounds of the arm. The additional functions P, B, and W are defined below.

Note that adapting this formulation to a different base kinematics is straightforward, as it only requires adapting the model (1) and the set of unknowns of the optimization problem. For instance, in the case of a differential drive base, $\dot{b}_y = 0$ in (1) and $\dot{b}_y$ should be removed from the set of unknowns.

The function P is defined as a logarithmic barrier at the joint angle limits:

$$P(\dot{\Theta}) = \sum_{i=1}^{N} P_i(\dot{\theta}_i) \quad (4)$$

$$P_i(\dot{\theta}_i) = \begin{cases} f = \theta_i + \dot{\theta}_i \Delta t & \\ -\lambda_i \log(f - \theta_i^{Min}) + \beta_i & , \quad \theta_i < \theta_i^L \\ 0 & , \quad \theta_i^L \leq \theta_i \leq \theta_i^H \\ -\lambda_i \log(-f + \theta_i^{Max}) + \beta_i & , \quad \theta_i > \theta_i^H \end{cases} \quad (5)$$

where λ_i is a scaling factor for the logarithmic barrier, β_i is a constant to assure continuity of the function, Θ_i^{Min} and Θ_i^{Max} are the angle limits of the i-th joint, and Θ_i^L and Θ_i^H are the repulsion starting points.

The function B is a penalty term avoiding the base to orient opposite to the goal. It is also a logarithmic barrier, defined by

$$B(\gamma) = -\lambda_B \log(-|\gamma| + 180) \quad (6)$$

where γ is the angle (in degrees) between the goal position and front side of the robotic platform, calculated as follows:

$$\gamma = \text{atan2}(y_g - \dot{b}_y \Delta t, x_g - \dot{b}_x \Delta t) - \dot{b}_\Theta \Delta t \quad (7)$$

for $P_g = (x_g, y_g, z_g)$ being the target goal position, and λ_B is a constant to scale the barrier.

Finally, function W is a regularizer over the actuation, defined by a weighted L_1 norm of the joint velocities

$$W(\dot{\Theta}, \dot{b}_x, \dot{b}_y, \dot{b}_\Theta) = \sum_{i=1}^{N} w_i |\dot{\theta}_i| + w_x \dot{b}_x + w_y \dot{b}_y + w_\Theta \dot{b}_\Theta \quad (8)$$

where $(w_1, \ldots, w_N, w_x, w_y, w_\theta)$ are the weights.

C. Implementation Details

The implementation is based on the well known middleware ROS (Robot Operating System). The Jacobian matrix was obtained using the open-source package PyKDL[3], which loadw A kinematic chain from an URDF[4] file, to then compute the Jacobian matrix based on the current joint state.

For numerical optimization we used the open-source package SciPy[5]. SciPy is an open source software package providing various numerical algorithms, namely optimization solvers. In this work, we use the Sequential Least SQuares Programming (SLSQP) constrained optimization solver.

IV. Experimental Evaluation

A. Real robot

For the real robot experiments, we used the Mbot robot [4] extended with a Robai 7-DOF Cyton Gamma 1500 Arm ($N = 7$). The base platform has an omnidirectional kinematics. To close the loop between perception and manipulation modules we use a marker detection off-the-shelf component. For this purposes, we used the camera located on top of Mbot head and a marker was printed and placed on top of the object to be grasped.

The parameters used in the experiments were: sampling period $\Delta t = 0.1$s, joint velocity limits $\dot{\Theta}^{Max} = -\dot{\Theta}^{Min} = (0.2, \ldots, 0.2)$, base velocity limits $\dot{b}^{Max} = -\dot{b}^{Min} = (0.3, 0.3, 0.2)$, barrier parameters

[2]Since the argument is a rotation matrix, the logarithm can be computed using the Rodrigues' rotation formula.

[3]http://wiki.ros.org/python_orocos_kdl

[4]http://wiki.ros.org/urdf

[5]https://www.scipy.org/

(a) Candle

(b) Pregrasp

Fig. 4: Starting arm configurations.

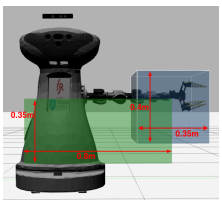
$\theta^H = -\theta^L = (0.1, \ldots, 0.1, 2.5)$, $\lambda_i = 0.25$ and $\beta_i = 0.1$ for $i = 1, \ldots, 7$, and $\lambda_B = 1.25$, and regularizer parameters $(w_1, \ldots, w_7, w_x, w_y, w_\theta) = (0.03, 0.025, 0.025, 0.02, 0.15, 0.01, 0.001, 4, 4, 8)/100$.

Due to the low dimensionality of the optimization problem, the solver was able to run in a significantly lower time than the sampling period Δt in common computers, thus allowing real-time execution.

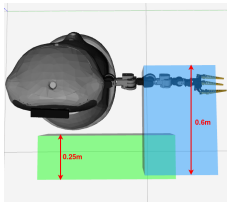
We performed 30 experiments by using only the arm and 20 for base + arm combined. We varies both the initial arm configuration (see Fig. 4), and the target random pose, randomly generated between specific bounds.

B. Simulation

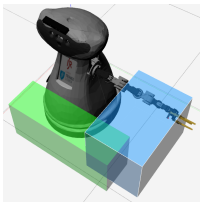
For simulation we used the Gazebo v7 physics simulator configured with ROS Control. For simulation testing purposes we ran the algorithm that generates random goal poses. In the test case when base movement is not required, we predefined area of generated poses, to increase the probability of pose being feasible (Fig. 5).



(a) Front view



(b) Top view



(c) Side view

Fig. 5: The volume in which the goal poses were generated.

The predefined areas have the following dimensions:

	Front area (green)	**Side area (blue)**
Dimensions [m]:	0.25 x 0.8 x 0.35	0.8 x 0.35 x 0.4
Distance from ground [m]:	(Z-axis) 0.3	(Z-axis) 0.35
Distance from baselink [m]:	(X-axis) 0.35	(Y-axis) 0.45

The decision on the dimensions of the random randomly generated pose areas was based on the robot use case scenarios. The realistic use case scenario for this particular robot is placing or fetching objects from the small table or chair, therefore our experiments were adjusted to these requirements.

C. Static base

The first experiment carried out in the simulation environment was based on random (but realistic) target poses within the predefined position area, not requiring base motion (Fig. 5). In this experiment base motion is disabled and we use only arm control. Taking into consideration that the developed controller is not aiming to reach the final grasping pose, but is targeting the pregrasp pose (the output of the grasp planning module), we set the following errors tolerances:

$$\text{Position error}: \quad \sqrt{e_x^2 + e_y^2 + e_z^2} = 0.01m \quad (9)$$

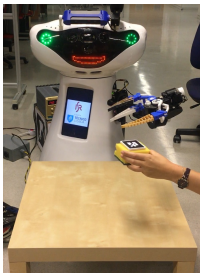
$$\text{Orientation error}: \quad \sqrt{e_r^2 + e_p^2 + e_y^2} = 0.05rad \quad (10)$$

where:
e_x, e_y, e_z - error between current position and goal position accordingly on X,Y,Z axes
e_r, e_p, e_y - error between current orientation and goal orientation accordingly on roll, pitch, yaw Euler angles

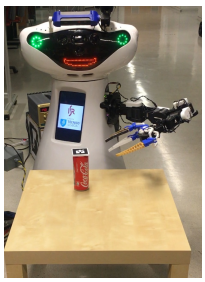
In the experimental part with realistic orientation of the generated pose, we restricted the roll and pitch rotation limits. The limitation of the yaw rotation was increased in comparison to the suggested reasonable limitations as followed:

roll axis	pitch axis	yaw axis
between 0°and 0°	between -3°and 3°	between -90°and 90°

In the experiments with the real robot, the goal pose comes from the robot perceptual module. In these experiments, we close the loop between perception and manipulation, by using head camera of the Mbot for marker detection, thus obtaining the goal pose. Due to pregrasp planner module



(a) Goal pose above the table



(b) Goal pose on the table

(c) Goal pose on the floor

Fig. 6: Different goal poses on real robot testing.

being under development process, we used fixed realistic front grasp orientation[6] for this test. Position of the goal pose is random (Fig. 6). For the real robot evaluation we increased the error tolerance for the following two reasons:

1) In simulation we have an ideal synthetic static pose, which is published at constant rate. For the real robot

[6]Goal orientation is given with respect to the center of the robotic platform, not to the end effector

we have used an Alvar marker detection algorithm, which position publication rate is unsteady. We recall that real world marker detection is subject to image noise, blur, and network delay. Because of that we have observed in our experiments an unstable (shaky) target pose.

2) Real robot manipulator joints suffer from backlash, and noisy encoder angle feedback which translates into end effector frame uncertainty once forward kinematics is computed. For that reason end effector frame was also presenting an unsteady behavior.

The real robot increased error tolerance as followed:

$$\text{Position error} \quad \sqrt{e_x^2 + e_y^2 + e_z^2} = 0.05m \quad (11)$$

$$\text{Orientation error} \quad \sqrt{e_r^2 + e_p^2 + e_y^2} = 0.1rad \quad (12)$$

Experiment summary:

Environment:	simulation	simulation	real robot
Goal position:	random	random	random
Goal orientation:	random	rand. realistic	fixed[7]
Grasp type:	front/side	front	front
Position error tolerance:	0.01m	0.01m	0.05m
Orient. error tolerance:	0.05rad	0.05rad	0.1rad
Starting arm config.:	candle	pregrasp	pregrasp
Total number of attempts:	100	50	20
SUCCESS RATE:	**78%**	**98%**	**80%**

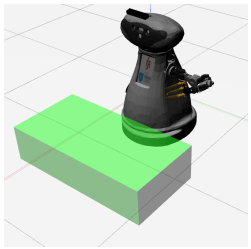
D. Discussion

Regarding the simulation results, it was expected that in some of the cases the controller gets stuck in the local minima configurations. Another issue is the timeout set to 120 secs. In the first test case scenario, poses are randomized in both position and orientation. For some of the extremely not convenient and unrealistic goal poses, the controller needs more than 120 sec to achieve the goal, that resulted in an higher failure rate than in the case of the second experiment, where the poses were restricted to reasonable rotational variations. On the other hand, while the human orders the robot to perform grasping object from the table, we assume that one expects the execution time to no more than 2 minutes.

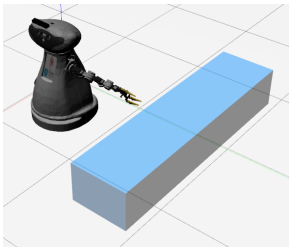
Considering the real robot testing, we encountered issues which do not apply in simulation environment, such as arm calibration errors and unreachable goal poses. Since the goal pose is not generated but perceived from the vision, testing algorithm no longer validates if the perceived goal pose is reachable. The last problem that caused 50% of failures on the real robot is the imperfection of perceptual module and markers detection. While the robot performs the task and gets closer to the goal pose, the end effector hardware part covers marker and the pose is no longer visible for the controller. As the second part of the real robot experiment we tested the arm following a moving object in real time, which results can be seen on the companion video[8].

E. Moving base

In this part of the experiments, base motion was required to reach the goal pose. We defined the area of generated poses in such a way to ensure that randomized pose is not reachable without moving the platform (Fig. 7). In this experiment we considered only randomized realistic orientations of the goal pose.



(a) Front grasp



(b) Side grasp

Fig. 7: Volume for randomized poses for base + arm.

The predefined areas have the following dimensions:

	Front area (green)	**Side area (blue)**
Dimensions [m]:	0.5 x 1.0 x 0.25	2.0 x 0.5 x 0.35
Distance from ground [m]:	(Z-axis) 0.35	(Z-axis) 0.35
Distance from baselink [m]:	(X-axis) 0.8	(Y-axis) 1.0

For the real robot testing the orientation of the goal pose was fixed during the whole experiment and the the error tolerance was increased due to hardware issues.

Experiment summary:

Environment:	simulation	real robot	real robot
Goal position:	random	random	random
Goal orientation:	rand. realistic	fixed[9]	fixed[10]
Grasp type:	front/side	front	side
Position error tolerance:	0.01m	0.05m	0.05m
Orientat. error tolerance:	0.05rad	0.1rad	0.1rad
Starting arm config.:	pregrasp/candle	pregrasp	candle
Total number of attempts:	100	20	20
SUCCESS RATE:	**100%**	**90%**	**75%**

During the real robot evaluation, we recorded video images from 3 different perspectives: rviz visualization of the real world, the image from the head camera used for markers detection, and a lab live camera placed overhead the laboratory, where the experiment was taken (Fig. 8).

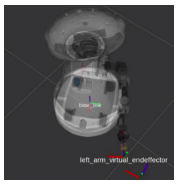
V. RESULTS

We define a *successful reach* of the goal pose once the pose error stabilizes within the specified tolerance. Altogether, we conducted 7 different experiments, 310 having reached the goal pose, which represents 89% of success rate (number of successful reaches over the total amount of attempts). Table I summarizes our results.

[8]Companion video showing a real robot using the optimization controller: `https://www.youtube.com/watch?v=_-M7cxlhyYY&t=2s`

(a) Live university camera image

(b) Mbot head camera image



(c) Rviz real world image

Fig. 8: Video data captured while conducting experiment.

		Success rate	
		arm	*arm and base*
Simulation	*front*	98%	100%
	side	–	100%
Real robot	*front*	80%	90%
	side	–	75%

TABLE I: Experimental evaluation results.

A. Discussion

The first experiment conducted on entirely randomized goal poses was predictably less successful than following experiment with a reasonable orientation of generated goal poses. The arm controller experiment on the real robot was less successful than expected results, based on the simulation success rate. Using real robot entails hardware issues, not modeled in simulation environment.

Concerning the experimental evaluation in the simulation environment, the success rate was 100%. Since the robot is able to move the base to the convenient configuration for grasping, not reachable poses are no longer existing. Being able to control the arm along with the base, not only shorten execution time but also significantly decrease the possibility of a goal pose not being feasible.

We observe a decline in performance when testing in the real robot, due to three issues: i) incorrect arm calibration, ii) unreliable marker detection, and iii) unreachable goal pose. Failures caused by inaccurate calibration were revealed by running Rviz real-world representation, along with the real arm execution. In the simulation software, the robot reached the goal pose, while that in the real world the end effector was slightly shifted with respect to the marker, due to arm calibration bias. Another issue was occlusion of the marker by the robot body. By observing the Rviz real-world representation we could easily observe that the goal pose some times disappears from the vision, which results in absense of a goal pose. One possible solution for this problem is to include an object tracking algorithm that is robust to occasional misdetections of the target. This problem of losing the marker detection affected negatively our experiments, since it increased the failure rate in the side grasping real robot experiment. Because of the starting arm configuration, the probability of covering the marker with the manipulator physical part was higher than in the front grasp task. Another issue is limited camera vision. In most of the cases, we were able to turn the robotic head to keep the marker in the vision. The issue encountered during side grasp tests when the starting head position is turned into the direction of the goal pose, while it is already close to its limits on the left side. A last observed issue, also contributing to a higher failure rate on the real robot testing, was missing a pregrasp planning module. In order to bypass this problem, we hardcoded reasonable realistic orientation for the end effector.

The obtained results show that simulation evaluation is usually more successful than the real robot testing results. We can also conclude that using the full arm and base controller yields higher probability of reaching the goal pose, than the arm controller alone. Another conclusion concerns the type of the grasp. When considering the integration of the controller with the perception module based on marker detection, the front grasping is more reliable than the side grasping.

B. Use case

SocRob@Home is a human and robot team targeting scientific robot competitions [10], such as the RoboCup@Home[11] and the European Robotics League[12]. It is well known that scientific robot competitions are a challenging evaluation environment, where robustness of the solutions is key. In fact, it was the lack of robustness of off-the-shelf manipulation solutions, such as the MoveIt[13], that originally pushed the authors to find the solution proposed in this paper. The implemented solution was integrated into the software architecture used by the team and it was successful used in the 3rd–4th place bronze cup finals of RoboCup@Home 2018, in Montreal.

VI. Conclusions

In this paper we have presented a solution for mobile manipulation controller for a combined arm and base platform system. We developed a real-time closed-loop Cartesian controller based on an constrained optimization approach. It is formulated in local coordinates and in velocity space.

The solution was evaluated quantitatively both in simulation and on a real robot. Moreover, our solution is fully integrated into ROS framework, and the source code was made openly available.

Future work includes several avenues, namely: (1) real-time obstacle avoidance by augmenting the optimization problem with motion constraints, and (2) robustness to calibration issues by tracking the end effector with the camera.

Acknowledgement

This work was supported by the FCT projects [UID/EEA/50009/2013] and [PTDC/EEI-SII/4698/2014], and by the German Federal Ministry of Education and Research (BMBF) under grant FKZ01IW180003.

[11] http://www.robocupathome.org/
[12] https://www.eu-robotics.net/robotics_league/
[13] https://moveit.ros.org/

REFERENCES

[1] François Chaumette and Seth Hutchinson. Visual servo control. i. basic approaches. *IEEE Robotics & Automation Magazine*, 13(4):82–90, 2006.

[2] S. Chitta, E. G. Jones, M. Ciocarlie, and K. Hsiao. Mobile manipulation in unstructured environments: Perception, planning, and execution. *IEEE Robotics Automation Magazine*, 19(2):58–71, June 2012.

[3] Matei Ciocarlie, Kaijen Hsiao, Edward Gil Jones, Sachin Chitta, Radu Bogdan Rusu, and Ioan A. Şucan. *Towards Reliable Grasping and Manipulation in Household Environments*, pages 241–252. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[4] João Messias, Rodrigo Ventura, Pedro Lima, João Sequeira, Paulo Alvito, Carlos Marques, and Paulo Carri co. A robotic platform for edutainment activities in a pediatric hospital. In *Proceedings of the IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 193–198, 2014.

[5] M. Nieuwenhuisen, D. Droeschel, D. Holz, J. Stückler, A. Berner, J. Li, R. Klein, and S. Behnke. Mobile bin picking with an anthropomorphic service robot. In *2013 IEEE International Conference on Robotics and Automation*, pages 2327–2334, May 2013.

[6] Jorg Stuckler, Ricarda Steffens, Dirk Holz, and Sven Behnke. Efficient 3d object perception and grasp planning for mobile manipulation in domestic environments. *Robotics and Autonomous Systems*, 61(10):1106 – 1115, 2013. Selected Papers from the 5th European Conference on Mobile Robots (ECMR 2011).

[7] J. Stückler and S. Behnke. Integrating indoor mobility, object manipulation, and intuitive interaction for domestic service tasks. In *2009 9th IEEE-RAS International Conference on Humanoid Robots*, pages 506–513, Dec 2009.

[8] I. A. Sucan, M. Moll, and L. E. Kavraki. The open motion planning library. *IEEE Robotics Automation Magazine*, 19(4):72–82, Dec 2012.

[9] Ioan A. Şucan and Lydia E. Kavraki. *Kinodynamic Motion Planning by Interior-Exterior Cell Exploration*, pages 449–464. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[10] Rodrigo Ventura, Meysam Basiri, André Mateus, João Garcia, Pedro Miraldo, Pedro Santos, and Pedro U. Lima. A domestic assistive robot developed through robot competitions. In *IJCAI Workshop on Autonomous Mobile Service Robots*, New York, 2016. IJCAI.