University of Verona

DEPARTMENT OF COMPUTER SCIENCE
Ph.D. degree in Computer Science

Ph.D. Thesis

Constrained Optimal Control for Semi-Autonomous Robotic Systems

Candidate:
Nicola Piccinelli

Thesis advisor:
Prof. Riccardo Muradore

Thesis submitted in 2022

Contents

1 Introduction **13**

1.1 Case study . . . 15

1.1.1 Robotic-assisted radical prostatectomy . . . 17

1.1.2 Laparoscopic radical nephrectomy . . . 18

1.2 Aims and objectives . . . 19

1.3 Outline . . . 19

2 Hardware setup **20**

2.1 The dVRK platform . . . 21

2.2 The *SARAS* platform . . . 22

2.2.1 Laparoscopic tools adapters . . . 23

2.2.2 Surgical tool adapter . . . 24

2.2.3 Force estimation using strain gauges . . . 25

2.3 Smart trocar . . . 29

2.4 Anatomical manikins . . . 31

3 Registration and calibration **35**

3.1 State of the art . . . 35

3.2 Multi arm 3D calibration . . . 38

3.2.1 Calibration accuracy evaluation . . . 42

3.3 Rigid 3D registration of pre-operative data . . . 48

3.3.1 Scaled ORB-SLAM . . . 49

3.3.2 Registration procedure . . . 51

3.3.3 Scaling evaluation and error bounds . . . 53

3.4 Qualitative and functional evaluation . . . 54

4 Supervisory control **56**

4.1 State of the art . . . 57

4.2 Modelling methodology . . . 59

4.2.1 Procedure depth . . . 61

4.2.2 Atomic and hierarchy-agnostic execution . . . 61

4.2.3 Event generation 61
4.2.4 Concurrent isolation 62
4.3 Case Study: Radical Prostatectomy 62
4.3.1 Observers 63
4.3.2 Surgemes 69
4.3.3 Left arm procedure 70
4.3.4 Right arm procedure 71
4.4 RARP execution and validation 73
4.4.1 Execution evaluation and analysis 77

5 Improved passivity-based bilateral teleoperation 79
5.1 State of the art 80
5.2 Novel energy tank dynamics 82
5.2.1 The bilateral teleoperation 87
5.2.2 Simulation comparison 90
5.3 Optimal modulation for passivity layer 93
5.3.1 Modulated tank based control 94
5.3.2 Optimised two-layer architecture 97
5.3.3 Experimental validation 99

6 Optimal control for safe bilateral teleoperation 103
6.1 State of the art 105
6.2 Robot model and environment estimation 107
6.2.1 Joint space feedback linearization 107
6.2.2 Cartesian space feedback linearization 108
6.2.3 Environment model 109
6.3 Passivity-based non-linear MPC 110
6.3.1 Environment torque estimation 111
6.3.2 Operator side teleoperation 112
6.3.3 Remote side teleoperation 114
6.3.4 Passive teleoperation on 1-DOF simulation 114
6.3.5 Passive teleoperation on 1-DOF robot 118
6.3.6 Force constraint on 6-DOF simulation 119
6.4 Force constraint using adaptive linear MPC 126
6.4.1 MPC with force constraint 129
6.4.2 Experimental results 133
6.5 Passivity-based adaptive linear MPC 138
6.5.1 Linear approximation of the tank dynamics 140
6.5.2 Adaptive linear MPC modelling 146

6.6 Passivity-based hybrid linear MPC 151
6.6.1 Linearized energy tank 152
6.6.2 Hybrid bilateral teleoperation 153
6.6.3 Hybrid MPC 155
6.6.4 Simulation results 156

7 Conclusion and Future Works 161

A libmpc++ 164
A.1 Linear MPC 165
A.2 Non-linear MPC 165
A.3 User manual 166
A.3.1 Allocation 166
A.3.2 Solver parametrization 166
A.3.3 Example of linear MPC 167
A.3.4 Example of non-linear MPC 169

B Author's publications 171

List of Figures

1.1 The concept schemes for the SOLO-SURGERY and LAPARO2.0-SURGERY platforms. 16

2.1 The platforms used for the experimental validation. a) the da Vinci® Research Kit (dVRK) available in the Altair Laboratory (University of Verona), b) the *SARAS* platform. 21

2.2 View of the Franka Emika manipulators inserted in the phantom's belly through the trocars. 22

2.3 CAD model of the *SARAS* tool adapter's base 24

2.4 CAD model of the motorized tool adapters. a) the adapter used for the actuation of the grasper and scissor tools, b) the adapter used for the actuation of the clip applier. 25

2.5 Mouting of the strain gauges on the grasper tool. a) a single strain gauge applied on the laparoscopic tool before being covered with protective material, b) three strain gauges applied on the *SARAS* laparoscopic tool after being covered with the protective material. 27

2.6 Linear ratio between measured and computed forces on the x and y axes. 28

2.7 CAD of the smart trocar prototype. a) a close view of the base plate, b) a complete view with the tool and trocar mounted on it. 30

2.8 The smart trocar's disk used to estimate the insertion and rotation values of the surgical instrument. 31

2.9 The 3D printed final prototype of the smart trocar used during the experimental validation; the trocars are mounted on a pair of adjustable arms to adapt to the anatomy; the diameter of the smart-trocar cannula is 5 mm. 32

2.10 The experimental validation of the proposed prototype: the solid line shows the laparoscopic tool position and the dashed line shows the reference position of the landmarks. 32

2.11 The anatomical manikins used during the evaluation of the RARP and the LRN. a) the da Vinci® robotic tools, the *SARAS* robotic tools and the phantom used during an experimental validation, b) the phantom is composed of bladder (1), rectum (2), urethra (3), prostate (4), seminal vesicles (5), fat (6), c) the external view of the abdominal phantom, d) the kidney model comprises the renal artery (1), renal vein (2), gonadal vein (3) and the ureter (4). 34

3.1 The reference frames produced by our proposed method (the axes direction of the reference frames are only for visualisation purpose). The orange transformations are known, whereas the black transformations are to be estimated. 39

3.2 The calibration components and the experimental setup. a) the calibration board with the marker, the coloured axes represents the common reference frame directions b) the adapter for the ECM positioning, c) the proposed setup for calibration, with the RealSense d435, the PSMs and the calibration pattern. . . . 42

3.3 Experimental setup for the calibration accuracy evaluation. a) setup for the localization and grasping experiment, the numbers on calibration board represents the nine grasping locations, b) the dual arm manipulation experiment consists in two arms carrying a ring while performing circular trajectories through the workspace. 43

3.4 The measured 3D positioning errors between the end-effector and the grasping point during the localization and grasping experiment. The error bars shows the mean and standard deviation across the nine grasping locations. 44

3.5 Absolute error of the dual arm manipulation through the workspace. The workspace has been projected using the Lambert equal-area cylindrical projection, the error is reported in mm. a) the workspace surface of the sphere with radius 10 mm, b) the projected surface of the sphere with radius 10 mm, c) the workspace surface of the sphere with radius 20 mm, d) the projected surface of the sphere with radius 20 mm, e) the workspace surface of the sphere with radius 30 mm, f) the projected surface of the sphere with radius 30 mm, g) the workspace surface of the sphere with radius 40 mm, h) the projected surface of the sphere with radius 40 mm. 45

3.6 Spiral-shaped trajectory executed by the PSM1 with our method in (a) and Tsai's method in (b). The red trajectory represents the kinematics of the PSM1, while the blue trajectory represents the marker identified in 3D space. 47
3.7 An example of re-projection of da Vinci® surgical instruments by using kinematic re-projection of the model directly onto camera color image. 47
3.8 The proposed software architecture for the 3D rigid registration and waypoint generation using a monocular camera. 49
3.9 The anatomical model extracted from the MRI. a) the segmented 3D model, b) the final point cloud used in the registration phase. 50
3.10 The resulting registration of the pre-operative map with the anatomy 3D reconstruction. The purple dots are part of the point cloud obtained by SLAM, the initial map is shown in green. 54
3.11 An example of the bladder pushing phase. a) the red sphere represents the point selected directly on the MRI, in this case the apex of the bladder, b) the robot initial position, c) the robot end-effector once it has reached the approach position over the bladder, d) the robot end-effector at the target position. 55

4.1 The graphical representation of a statechart. a) an example of a two-level and concurrent statechart, b) the revised statechart is composed of n observers that run concurrently to the procedure. The observers generate events consumed by the procedure's state transitions. The procedure hierarchy is not reported in this figure but is composed of three levels. 59
4.2 Left arm procedure modeled as statechart (i.e., bladder neck transection). The observer states are represented by O_i, the surgeme states are represented by S_j and the observed events and the triggers are labeled with o_k and t_k, respectively. 64
4.3 Right arm procedure modeled as statechart (i.e., bladder mobilization, vesicourethral anastomosis). The observer states are represented by O_i, the surgeme states are represented by S_j and the observed events and the triggers are labeled with o_k and t_k respectively. 65

4.4 A series of snapshots taken from the experimental validation during the *bladder neck transection*. State transitions are indicated in the captions as arrows and since some of them occurs quickly the transitions are sometimes grouped together in the same snapshot. A detailed view of the transitions occurrences is available in Figure 4.5. 74
4.5 The sequence diagram shows the time when each transition occurred during the *bladder neck transection*. The time flow is represented by the vertical dashed lines. A transition that occurred between two states is indicated as an arrow, the label above the arrow indicates the trigger which activates the transition. On the right side of the diagram, the transition times are reported. The red, blue, and black colours are used to group the transitions occurring for phases, actions and surgemes respectively. 75
4.6 The catheter speed and the modulus of the interaction force during the *follow* and *grasp* action respectively. a) the horizontal black dashed lines represent the *feature observer* threshold, the vertical black lines represent the action transition and the vertical dotted black lines represent the surgeme transitions, b) the horizontal black dashed line represents the *force observer* threshold, the vertical black lines represent the action transitions and the vertical dotted black lines represent the surgeme transitions. 76

5.1 The control scheme of the bilateral teleoperation with the novel tank dynamic. 86
5.2 Time evolution of the classical tank dynamics. In red and blue the local and remote side tank respectively. a) the evolution of the tank, b) the power dissipated by the robots, c) the power generated by the control action, d) the output power. 91
5.3 Time evolution of the novel tank dynamics. In red and blue the local and remote side tank respectively. a) the evolution of the tank, b) the power dissipated by the robots, c) the power generated by the control action, d) the output power. 92
5.4 The control architecture for a robot 94
5.5 The optimized two-layer architecture for bilateral teleoperation . 97

5.6 The experimental validation using a torque controlled manipulator (data are reported only along the x-axis). a,b) the Cartesian position and velocity of the local and remote manipulators, c) the local side force feedback and the remote side interaction force, d) the energy evolution of the local and remote side tank. 100
5.7 Energy requests and power flows between the energy tanks. In dotted blue line the local-to-remote energy flow and in dashed red line the vice versa. 100
5.8 The experimental validation using a surgical robotic platform (data are reported only along the x-axis). a,b) the Cartesian position and velocity of the local and remote manipulators, c) the local force feedback and the remote interaction force, d) the energy evolution of the local and remote side tank. 101

6.1 The bilateral teleoperation control architecture. The green blocks are the P-MPC controllers, the blue blocks are the sources of external torques, the grey blocks are the robots, the orange blocks are the tank estimators and the dashed line is the communication channel where P^{out} and E^{req} are exchanged. 111
6.2 Two-layers performance during the simulated soft contact. The blue and orange lines are the operator and environment robot trajectories, respectively. a) two-layer position tracking, b) two-layer velocity tracking, c) two-layer torque commanded, d) two-layer stored energy. 116
6.3 P-MPC performance during the simulated soft contact. The blue and orange lines are the operator and environment robot trajectories, respectively. a) P-MPC position tracking, b) P-MPC velocity tracking, c) P-MPC torque commanded, d) P-MPC stored energy. 117
6.4 A magnification of the environment side tank behaviour during the simulated comparison and the experimental setup. The blue and orange lines are the environment tank evolution for the P-MPC and the two-layer approaches, respectively. The time axis and the data are the same of Figure 6.3 and Figure 6.2. 117
6.5 The experimental teleoperation setup. (1, 2) DC motors with encoder and gearbox, (3) motors controller, (4) environment side obstacle. 118

6.6 Hard contact with constant delay P-MPC. The blue and orange lines are the operator and environment robot trajectories respectively. a) position tracking, b) velocity tracking, c) torque commanded, d) stored energy. 120
6.7 Hard contact with variable delay P-MPC. The blue and orange lines are the operator and environment robt trajectories respectively. a) position tracking, b) velocity tracking, c) torque commanded, d) stored energy. 121
6.8 Simulation setup. The blue and the yellow ring shows the operator and remote side robot's end-effector, respectively. 122
6.9 Simulation results for the soft contact scenario with no communication delay. a) Cartesian position along x, y, z axes (red, green and blue lines respectively) of the operator (solid line) and remote (dashed line) side robot, b) command torque along x, y, z axes of the operator and remote side robot, c) estimation of environment damping along x, y, z axes, d) estimation of the environment stiffness along the y axis, e) modulus of the interaction force of the remote side robot, d) energy tanks at the operator and remote sides. 124
6.10 Simulation results for the soft contact scenario with constant communication delay. a) Cartesian position along x, y, z axes (red, green and blue lines respectively) of the operator (solid line) and remote (dashed line) side robot, b) command torque along x, y, z axes of the operator and remote side robot, c) estimation of environment damping along x, y, z axes, d) estimation of the environment stiffness along the y axis, e) modulus of the interaction force of the remote side robot, d) energy tanks of the operator and remote side robot. 125
6.11 The block diagram of the proposed control architecture; in light blue the manipulator dynamics and the inner joints controller. The control torque $\boldsymbol{\tau}$ within the dotted box is exclusively managed by the robot driver. 129
6.12 The UR5e, equipped with a 3D printed tool, over the environment surface used in the experimental validation. The orange plastic sheet covers the four compliant materials as numbered in the picture. 133

6.13 Cartesian and joint space robot's trajectories during the 95 N experiment. a) the trajectory and the measured position of the robot's end-effector along the x,y,z axes. The red dashed line is the estimated position of the environment $\boldsymbol{x}_e$, b) the reference joint position $\boldsymbol{q}_r$ computed by the feedback linearization block shown in Figure 6.11. 136
6.14 Experimental validation for different force limits: 95 N and 75 N; (a),(b) environment forces along the z axis: the blue line is the measured force, the orange line is the estimated force using the RLS, and the dashed red line is the force limit; (c),(d) the estimated environment stiffness along the z axis; (e),(f) MPC optimal command wrench along the x,y,z axes. 137
6.15 Comparison between the non-linear tank dynamic (in red) and the tank linearization (in blue). a) the naive linearization, b) the linearization under the proposed constraints. 141
6.16 The MPC-based control architecture. The green blocks are the control modules, the blue blocks are the external torque sources, the grey blocks are the manipulators and the light grey is the communication channel. 147
6.17 The MPC-based control architecture. The green blocks are the H-MPC control modules, the blue blocks are the external torque sources, the grey blocks are the manipulators and the light grey is the communication channel. 153
6.18 The finite state machine representing the jump dynamics of the hybrid bilateral teleoperation. 154
6.19 Simulation of soft contact with force constraints under constant communication delay. a) in black the desired trajectory, and in red and blue the position of the operator- and remote-side manipulators, respectively, b) command torques at the operator side (red) and at the remote-side (blue), c) switches of the contact state over time, d) the reaction environment force, in dashed black the upper and lower bounds. 158
6.20 Simulation of hard contact with bounce reduction under constant communication delay. a) in black the desired trajectory, and in red and blue the position of the operator- and remote-side manipulators, respectively, b) command torques at the operator side (red) and at the remote-side (blue), c) switches of the contact state over time, d) the reaction environment force. . . . 159

6.21 Effect of the bounces reduction. a) the chattering on the command input with $\lambda = 0$, b) the chattering effect on the command input with $\lambda = 1e^{-6}$, c) the amount of contact state transitions in function of the λ weight of the cost function. 160

List of Tables

2.1 RealSense d435 specifications 21

3.1 A comparison of the error in the localization and grasping test . 43
3.2 The positioning error between the PSM1 and PSM2 during the dual-arm manipulation experiment 46
3.3 A comparison of the error between the marker tip trajectory and the measured tip trajectory for the projection test 47
3.4 Overall system accuracy evaluation, the position of the points are expressed in $\mathbb{R}^3$. 53

4.1 Description of symbols for observers, surgemes and triggers of the left arm in Figure 4.2. 66
4.2 Description of symbols for observers, surgemes and triggers of the right arm in Figure 4.3. 67

6.1 RMSE and Percentage RMSE of the manipulator and controller estimated torques 134

Chapter 1

Introduction

An autonomous robotic system can make decisions and perform actions without direct control, handling real-world conditions that are unpredictable and dynamic. In recent years, the research interest in Robotic-assisted Minimally Invasive Surgery (R-MIS) is shifting from teleoperated devices to autonomous support systems for the execution of repetitive surgical steps (i.e., suturing, ablation and microscopic image scanning [1]). Even if teleoperation is widely used (see e.g., the da Vinci® by Intuitive Surgical [2] or the DLR MiroSurge [3]) for improving the quality of the operation and for enabling remote surgery [4], [5], the workload of surgeons is very high and there is room to further improve the efficiency of the medical system.

A higher level of autonomy can potentially improve the quality of an intervention in terms of patient safety and recovery time [6]. Moreover, it can optimise the use of operating rooms, increase the number of surgeries without affecting the surgeon's workload and therefore reducing the hospital costs. In general, autonomy requires systems with advanced capabilities in perception, reasoning, decision making [7], motion planning [8]–[10] and interaction with the physical environment. Nonetheless, for autonomous or semi-autonomous systems human-robot interaction plays a key role in providing both safety of execution and a successful knowledge transfer between users and robots. Apart from the performance of task fulfilment, one major concern in robotic surgery is safety and this requirement becomes of paramount importance in the case of autonomous and/or semi-autonomous surgery [11]. In this setup, as we will see later in this thesis, optimal control and passivity theory will be exploited to design a framework able to embed constraints and guarantee performance and safety.

Generally speaking, optimal control aims to find a solution to dynamic optimization problems by computing a vector of control inputs such that an

objective function is minimized over some time horizon [12]. Recent technological advances in computer science have dramatically decreased the costs and increased the capability of computers and sensors, making the adoption of computationally intensive control systems more appealing. At the same time, advances in the mathematics and computational algorithms for optimization have greatly improved the speed and reliability of the calculations required by the Model Predictive Control (MPC) used in this thesis [13].

Humans actually use an intuitive version of MPC in many aspects of their daily lives: it is, for instance, the way that you catch a ball or play chess. In fact, you use a prediction of the future to inform your present decision, as a good player does while playing chess. A good player tries to predict all the possible moves as far into the future as possible before making a move, but since it is impossible to extend these predictions to the game's end, the prediction horizon is finite and incomplete. From this analogy we can derive several facts, the first is that the prediction is only for a finite number of moves because the complexity of prediction increases very rapidly with the number of steps. Moreover, to decide the best move based on the prediction it is required a performance measure to evaluate the possible actions. Finally the more moves can be accurately predicted the best the outcome will be. So in general, the basic idea of the MPC is to use a mathematical model of the system to be controlled to predict its behaviour n time steps into the future (hence, model predictive) and the prediction depends on the specific control used. Based on this model, the MPC chooses the best sequence of controls by minimizing a cost function. Then, the first element of the such controls sequence is applied to the system. After this first control value is applied, the system responds, a new measurement is taken, and the MPC problem is solved again. The most common choice for the cost function is quadratic in both the control signal and the system states because energy in many situations is quadratic [13].

The most important results in controls were the Linear Quadratic Regulator (LQR) and the Linear Quadratic Gaussian Regulator (LQG). However, LQR completely ignores the actuator saturation [13]. Control engineer had long ago realized that actuator saturation is important and developed a theory of optimal control that could include the effects of saturation. This was the minimum principle proposed by Pontryagin and his colleagues. Unfortunately, the minimum principle had three important flaws even under the best of circumstances: it only provides an open-loop optimal control; it is difficult to compute the solution; the optimal control problem has to cover only a finite time except in some very special cases.

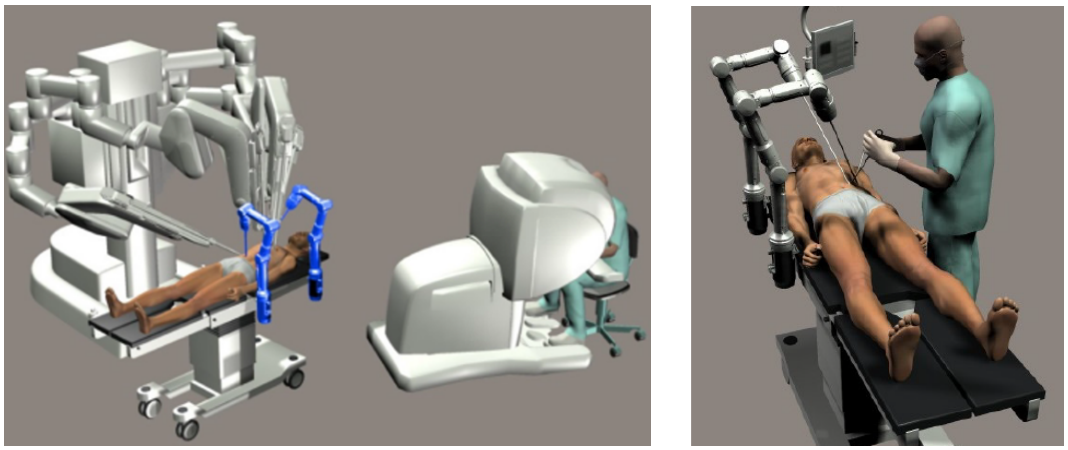
To solve the aforementioned optimal control problem issues it is common to

discretize the original problem in time and then use linear, quadratic, convex, or non-linear programming methods to solve the discrete-time problem. However, the open-loop optimal control heavily depends on the initial condition and the future state and control vectors, thus in practice, yields to the fact that feedback controls are far superior to even the best open-loop option. For this reason, a way to approximate the optimal open-loop controller by a closed-loop controller is to recompute the optimal control at each new discrete-time instant. In order to do this the future state and control had to be predicted after the time at which the optimal control problem is terminated. However, optimality over a finite time interval does not imply stability. This sets the stage for Model Predictive Control [13].

Passivity theory provides a modular approach for the design and analysis of control systems using an input-output description based on energy-related considerations. The main idea is that many important physical systems have certain input-output properties related to the conservation, dissipation, and transport of energy [14]. Robustness in controller design is an important problem which relates to how the closed-loop system will perform when the physical system differs either in structure or in parameters from the its mathematical model. If the stability of the controller is designed exploiting on the passive properties, the closed-loop system will be stable whatever the values of the physical parameters until the system is passive. Moreover, a system made of feedback and parallel interconnections of passive subsystems inherit the passivity property of the connected blocks. This simplifies the analysis by allowing for the manipulation of block diagrams and provides guidelines on how to design control systems (i.e., Passivity-Based Control, PBC). A further indication of the usefulness of passivity theory is the fact that the PID controller is a passive system, and a fundamental result that will be presented is the fact that the stability of a passive system with a PID controller can be established using passivity arguments. There is another aspect of passivity which is very useful in practical applications [14]. It turns out that passivity considerations are helpful for guiding the placement of sensors for feedback control. From a network modelling point of view, passive systems arise as port-Hamiltonian systems [15].

1.1 Case study

The approaches presented in this thesis have been mainly developed and tested within the European Research Project "Smart Autonomous Robotic Assistant



(a) SOLO-SURGERY (b) LAPARO2.0-SURGERY

Figure 1.1: The concept schemes for the SOLO-SURGERY and LAPARO2.0-SURGERY platforms.

Surgeon (*SARAS*)". *SARAS* aimed to develop a surgical robotic platform that allows a single surgeon to execute R-MIS operations. This would increase the social and economic efficiency of a hospital while guaranteeing the same level of safety for the patients. The robotic platform used in the project was made of two assistive robotic arms and has been designed to implement the tasks currently done by the assistant surgeon in the operating room. Such arms have been controlled through a cooperative and cognitive supervisor system able to infer the actual state of the procedure and to act accordingly with the surgeon's needs. During the *SARAS* project the following three platforms of increasing levels of complexity have been developed [1]:

MULTIROBOTS-SURGERY. In this platform the main surgeon uses a da Vinci® robot (by Intuitive Surgical Inc.) equipped with the da Vinci® Research Kit (dVRK 5) while the assistant surgeon teleoperates the *SARAS* assistive robotic arms using a haptic device with force feedback. This system has been used to collect the data needed to design the semi-autonomous *SARAS* control architecture.

SOLO-SURGERY. In this platform (see Figure 1.1a), the *SARAS* system was autonomously playing the role of the assistant to help the main surgeon at the da Vinci® console performing the surgical procedure. The da Vinci® console of the main surgeon has been enhanced by integrating

[1]The videos of the *SARAS* project are available online

force/tactile feedback and a speech recognition module to interact with the *SARAS* system.

LAPARO2.0-SURGERY. In this final platform (see Figure 1.1b), the *SARAS* system played the same role as in the SOLO-SURGERY case, but with the main surgeon using standard handheld laparoscopic tools. This last platform aimed to let the *SARAS* system be available to hospitals that cannot afford to buy a da Vinci® like surgical robotic system.

The validation of the *SARAS* concept focused on two specific and very important robotic-assisted procedures as they cover almost 80% of all robotic interventions. The first procedure is the *Radical prostatectomy* (RARP), which is the resection of the whole prostate gland in male patients with prostate cancer while preserving urinary continence and erectile function. The latter is the *Laparoscopic Radical Nephrectomy* (LRN) which is the removal of the renal cell carcinoma while keeping the healthy part of the kidney.

1.1.1 Robotic-assisted radical prostatectomy

Robotic-Assisted Radical Prostatectomy (RARP) is a surgical procedure where the surgeon utilises a robotic manipulator to remove the prostate along with, in some cases, the seminal vesicles and the pelvic lymph nodes [16]. The procedure is performed for the treatment of prostate cancer. All results of robotic prostatectomy so far indicate the benefits of minimally invasive surgery while also showing encouraging short and long-term outcomes in terms of continence, potency, and cancer control; it is regarded as a major innovation in the surgical treatment of prostate cancer. RARPs are currently performed using either the da Vinci® surgical system or any other comparable robotic platform. Surgeons remotely control the instruments of the robotic manipulator using two joysticks available on the console. In the operating room, there must be also an assistant surgeon next to the patient, helping the main surgeon.

The RARP-like procedure modelled in *SARAS* was presented in [17], [18] the first surgeon control with the da Vinci® console two instruments and an endoscope, while the assistant handles two standard laparoscopy instruments. At first, the surgeon identifies the proper plane of dissection to operate on the bladder neck, which is then divided transversely concerning the urethra, until he/she identifies the urethral catheter pushed through the prostate. At this point, the assistant surgeon, using the right laparoscopic tool, mobilizes the bladder to clear the view, and, with the left laparoscopic tool, raises the prostate. Once the prostate is suspended anteriorly, the main surgeon grasps

the tip of the catheter and lifts it upwards to increase access to the lower part of the prostate, including the vas deferens and the seminal vesicles. After the prostate removal, the main surgeon performs the vesicourethral anastomosis. During this phase, the activities of the assistant surgeon consist of avoiding the bladder inflation by keeping it pushed down and, once the suture has been completed, cutting the needle's thread with the scissors [16].

1.1.2 Laparoscopic radical nephrectomy

Laparoscopic Radical Nephrectomy (LRN) is a surgical procedure to remove (through 4-5 small incisions across the abdomen) the entire kidney. To do so, usually some additional structures are removed, such as part of the ureter that connects the kidney to the bladder or other adjacent structures such as the adrenal gland or lymph nodes. The use of laparoscopic renal surgery minimizes patient morbidity as well as improves patient outcomes compared to open renal surgery [19], [20]. The benefits with respect to the open surgery are in terms of reduction in the recovery time, hospital stay and postoperative complications. Different techniques have been used to perform Laparoscopic Nephrectomy, using either the trans-peritoneal or the retroperitoneal route. The choice of the laparoscopic approach is dictated by the location and the technical complexity of the renal mass. In this thesis we follow the current practice carried out at the San Raffaele Hospital [17], [18], partner of the *SARAS* project.

At first, the surgeon proceeds with the colon mobilization to expose the renal vein, the gonadal vessels and the ureter. The do so, the Gerota's fatty tissue is incised at the level of the lower pole of the kidney and the assistant surgeon helps the main surgeon by tractioning the Gerota's tissue. At this point, the ureter and gonadal vessels are identified and attachments between the psoas muscle and Gerota's fascia are released with sharp and blunt dissection with the monopolar scissors. Then, to proceed with the dissection of the renal hilum the renal artery and the renal vein are dissected. From now on, the assistant surgeon changes the tool using a clip applier to perform all the required dissections. Hem-O-lok clips are applied to the artery and the same clips are used on the renal vein. Following the division of the renal vein, the clipping of the renal artery is completed and it is then divided. Finally, the ureter is ligated with another Hem-O-lok clips and transected to allow the kidney to be fully mobilized.

1.2 Aims and objectives

This thesis aims to investigate the design of a supervisory controller for a semi-autonomous robotic system and the application of optimal control techniques to manage unexpected events and constraints. The management of such events will be demanded by the surgeon through a novel formulation of constrained bilateral teleoperation. For this reason, particular focus will be given to addressing MPC problems under input-output and state constraints for linear, non-linear and hybrid (i.e., with both continuous and discrete dynamics) systems. These constraints will be used to guarantee stability and enforce safety conditions during the teleoperated intervention. The thesis will also cover the research activities necessary to develop the *SARAS* semi-autonomous system. Such activities include the definition of a shared reference system for the multi-arm robotic setup and the registration of 3D preoperative medical data to the patient's anatomy.

1.3 Outline

The thesis is organised into six chapters and, besides the Chapter 1 and Chapter 2 where a general overview of the thesis and the hardware setup are presented, each of the remaining chapters will focus on a specific aspect of this study. The chapters are self-contained and organized in such a way to

- recall the relevant literature
- present the proposed solution
- discuss the simulation and/or experimental results
- draw some conclusion and the connection with the *SARAS* platform

Chapter 3 focuses on the basic requirements needed by the *SARAS* platforms, Chapter 4 presents the autonomy degree of the *SARAS* robotic systems, Chapter 5 proposes an improved version of a passive bilateral teleoperation architecture where optimal control is exploited to ensure the system stability under communication delay. Finally, Chapter 6 presents the MPC-based novel constrained bilateral teleoperation architecture, where the system stability requirement is reformulated as an optimization constraint together with other safety constraints, such as upper bounds on the interaction force at the end-effector of the robot at the remote side.

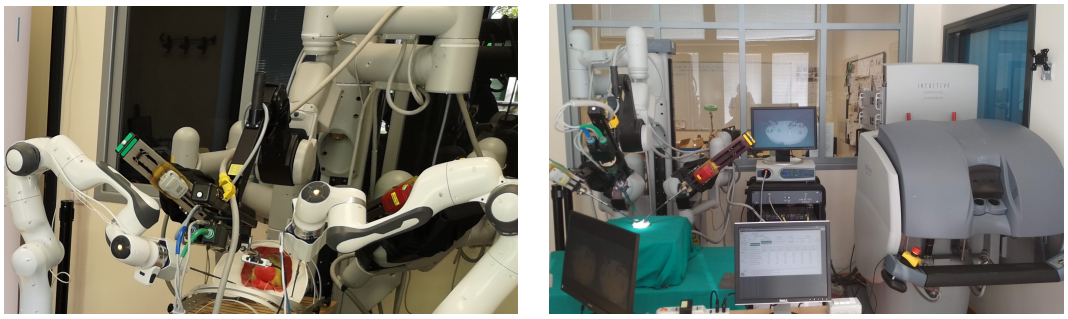
Chapter 2

Hardware setup

The experimental validation of the methodologies proposed in this thesis are carried out mostly in the surgical setup developed for the *SARAS* project. As mentioned in Chapter 1, the case study consists of the development of semi-autonomous robotic platforms for R-MIS and MIS. In the R-MIS platform called SOLO-SURGERY platform, the setup is composed of the dVRK together with the *SARAS* multi-arm robot. In the MIS platform called LAPARO2.0-SURGERY platform, the main surgeon performs the surgery using a pair of standard laparoscopic tools. In this last scenario, the lack of the da Vinci® robot requires to designed a novel tracking system to estimate the position and the orientation of the tools' tips that we called *smart trocar*. A trocar is a medical device made up of an awl, a cannula and a seal; they are placed through the abdomen during laparoscopic surgery. The trocar functions as a portal for the subsequent insertion of other instruments, such as graspers, scissors, clip applier, etc. In this chapter, we will provide an overview on the design and manufacturing of the smart trocar prototype together with a preliminary quantitative evaluation of the tracking accuracy. The surgical environment is made of anatomical realistic manikins, specific for each procedure. The phantoms are detailed models of the lower abdomen and the left upper abdomen for the RARP and LRN, respectively. The whole control system has been developed and integrated using the Robot Operating System (ROS).

Part of this chapter is based on the following publication:

› N. Piccinelli, E. Ghignoni, M. Piano, G. D. Rossi, F. Setti, and R. Muradore, "Prototyping a laparoscopic smart trocar for semi-autonomous surgical robotics," in *Conference on New Technologies for Computer and Robot Assisted Surgery (CRAS)*, 2022



(a) *SARAS* platform (b) dVRK platform

Figure 2.1: The platforms used for the experimental validation. a) the da Vinci® Research Kit (dVRK) available in the Altair Laboratory (University of Verona), b) the *SARAS* platform.

Table 2.1: RealSense d435 specifications

Camera specifications	
Resolution	1280×720
Field of view (FOV)	$91° \times 65° \times 100°$
Frame rate	90 fps
Baseline	50 mm
Z-accuracy	$\leq 2\%$ of the working distance

2.1 The dVRK platform

The idea behind the dVRK initiative is to provide the core hardware, i.e., a first-generation da Vinci surgical system, to a network of researchers worldwide, by repurposing retired clinical systems. This hardware is provided in combination with dedicated electronics to create a system that enables researchers to access to any level of the control system of the robot as well as the data streams within it. The dVRK components, shown in Figure 2.1b, are the master console (the interface at the surgeon side), the robotic arms to handle the tools and the endoscope at the patient side, and the controller boxes containing the electronics [22]. The robotics arms of the dVRK are called: Master Tool Manipulators (MTMs), Patient Side Manipulators (PSMs) and the Endoscopic Camera Manipulator (ECM) [23]. We augmented the stereo endoscope with an Intel RealSense d435 RGB-D camera rigidly attached to the endoscope through a 3D printed adapter. The camera specifications are reported in Table 2.1. The present setup is not compatible with a surgical scenario. However it is well possible that in the near future miniaturized RGB-D cameras could be integrated within the endoscope.

Figure 2.2: View of the Franka Emika manipulators inserted in the phantom's belly through the trocars.

2.2 The *SARAS* platform

The *SARAS* robotic platform, shown in Figure 2.1a, is composed of two Franka Emika's Panda robots used to automate the role of the assistant surgeon and three laparoscopic tool adapters. The adapters allow to use standard laparoscopic tools such as graspers, scissors and clip-appliers as custom end-effectors. Moreover, we also augmented our standard laparoscopic tool with strain gauges to estimate the interaction forces with the anatomy. The robots are constrained by a software remote centre of motion (RCM) mechanism embedded in a custom Cartesian position-controller. The RCM mechanism forces the surgical tools to pivot around the incision ports, where the trocar is inserted as shown in Figure 2.2, and avoids damage on the surrounding tissue. In literature there exists two main ways to preserve the RCM constraint: by mechanical design or by software. Even if mechanical RCM is highly precise and used in the da Vinci® robot, software RCM provides flexibility and versatility [24]. To compute the target homogenous transformation $\boldsymbol{T}_t(t)$ the controller takes as inputs the Cartesian target position $\boldsymbol{x}_t(t)$ and the target rotation angle $\gamma_t(t)$. Let $\boldsymbol{x}_r$ be the Cartesian position of the desired RCM point, the normal direction towards the target position $\boldsymbol{x}_t(t)$ can be computed as

$$\boldsymbol{n}(t) = \frac{\boldsymbol{x}_t(t) - \boldsymbol{x}_r}{|\boldsymbol{x}_t(t) - \boldsymbol{x}_r|_2}. \tag{2.1}$$

Let $\boldsymbol{d}$ be an arbitrary unit vector non-parallel to $\boldsymbol{n}(t)$. The rotation matrix for the end-effector satisfying the RCM constraint can be so defined as

$$\boldsymbol{R}_r(t) = \begin{pmatrix} \boldsymbol{r}_1(t) & \boldsymbol{r}_2(t) & \boldsymbol{r}_3(t) \end{pmatrix} \tag{2.2}$$

where

$$\begin{cases} \boldsymbol{r}_1(t) = \boldsymbol{n}(t) \times \boldsymbol{r}_2(t) \\ \boldsymbol{r}_2(t) = \boldsymbol{n}(t) \times \boldsymbol{d} \\ \boldsymbol{r}_3(t) = \boldsymbol{n}(t) \end{cases} \tag{2.3}$$

The rotational part of the target transformation $\boldsymbol{T}_t(t)$ can be now defined as the following sequence of rotations

$$\boldsymbol{R}_t(t) = \boldsymbol{R}_r(t)\boldsymbol{R}_i\boldsymbol{R}_{\gamma_t}(t) \tag{2.4}$$

where $\boldsymbol{R}_{\gamma_t}(t)$ is the desired tool axial rotation and $\boldsymbol{R}_i$ is the initial rotation offset matrix. This offset matrix is computed only the first time and is defined as

$$\boldsymbol{R}_i = \boldsymbol{R}_r^{-1}\boldsymbol{R}_e \tag{2.5}$$

where $\boldsymbol{R}_e$ is the measured orientation of the robot's end-effector. Finally the translational part of $\boldsymbol{T}_t$ is $\boldsymbol{t}_t(t) = \boldsymbol{x}_t(t)$

$$\boldsymbol{T}_t(t) = \begin{pmatrix} \boldsymbol{R}_t(t) & \boldsymbol{t}_t(t) \\ 0 & 1 \end{pmatrix}.$$

In this way, the controller drives the robot end-effector guaranteeing the RCM constraint.

When a new target position is received the controller computes a 7-th order spline in order to set positions, velocities, accelerations and jerks as boundary conditions. In our case the initial conditions are the current state of the manipulator's end-effector at the very first run, while later it uses the previous computed spline evaluated at the current time. When a new target point is received, the final conditions are set to zero, exception made for the position which is set as the desired position $\boldsymbol{x}_t$ and the desired rotation γ_t. The usage of a high-degree polynomial allows to achieve a smooth motion of the end-effector which guarantees continuities on position, velocity, acceleration and jerk profiles.

2.2.1 Laparoscopic tools adapters

SARAS requires to perform specific surgical tasks like cutting threads, clipping and interacting with tiny objects (i.e., in the order of few mm). Since the

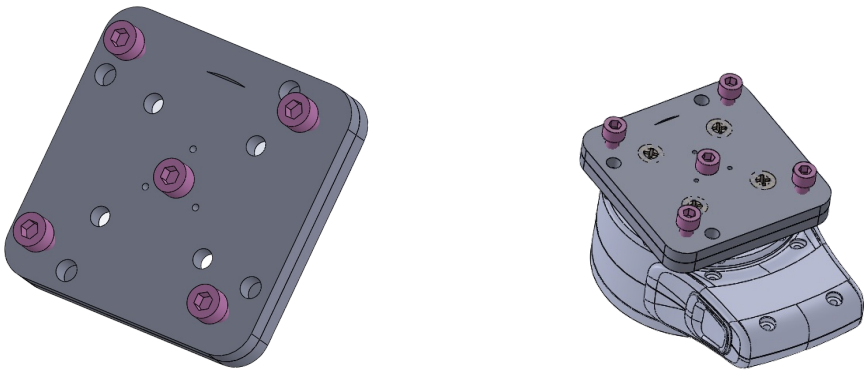


Figure 2.3: CAD model of the *SARAS* tool adapter's base

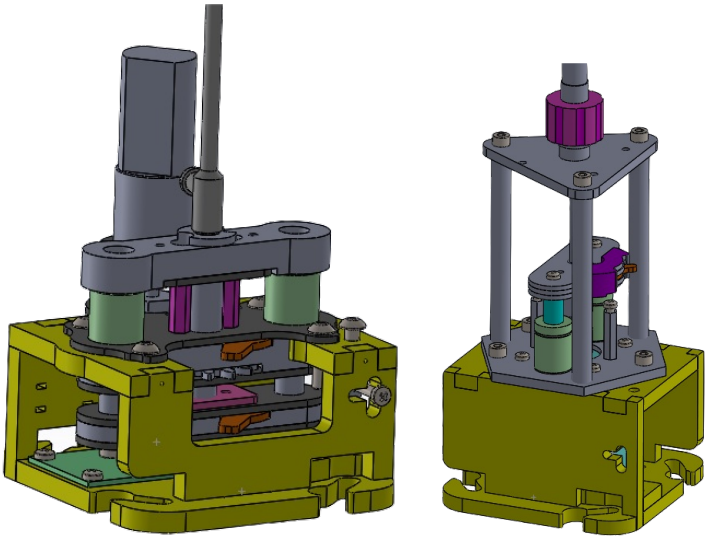
Franka Emika robot is shipped with a big industrial grasper we designed and manufactured a set of laparoscopic tool adapters. The adapters are composed of two parts: the first is the interface between the robot's flange and the adapter base, while the latter is the adapter itself. Figure 2.3 shows a close-up view of the flange-adapter interface. The base is also equipped with an easy release system that allows a quick replacement of the instruments during the procedure.

2.2.2 Surgical tool adapter

The adapter has been designed to hold a specific set of laparoscopic instruments and to provide the motorized opening and closing of their jaws. The tool rotation along the main axis is provided by the last joint of the Panda robot. The adapter's actuator is controlled by a dedicated board connected via USB to the control PC of the whole robot. The communication between the board and the PC is via serial port.

Grasper and scissor adapter

The adapter design for the grasper and scissor is the same, the actuator is placed aside from the tool shaft to reduce the height of the adapter as much as possible. The actuator is a servo DC motor and the actuation for opening and closing of the instrument is performed by moving a plate at which the internal open/close mechanism of the surgical instrument is attached. The adapter's board in this case is placed at the bottom of the adapter with a micro-switch which allows to manually open/close the tool in case of emergency. The adapter is finally covered using a 3D printed carter to avoid the direct contact with the moving parts and the electronic board. The 3D CAD model of the



(a) (b)

Figure 2.4: CAD model of the motorized tool adapters. a) the adapter used for the actuation of the grasper and scissor tools, b) the adapter used for the actuation of the clip applier.

aforementioned adapter is shown in Figure 2.4a.

Clip-applier adapter

The adapter design for the clip-applier is slightly different since the torque required to apply the clip is larger than the one needed for the closing the grasper and the scissor. For this reason we moved from servo DC motor to a stepper motor, and we placed the motor below the main axis of the tool. In this revised design, the adapter's board is not on the adapter itself but it's moved on a separate box to have access to the manual control also when the carter is mounted. The 3D CAD model of the aforementioned adapter is shown in Figure 2.4b.

2.2.3 Force estimation using strain gauges

To estimate the force exerted by the surgical instruments on the patient anatomy we applied a set of three strain gauges sensors along each tool shaft and we designed a linear regression to relate the strain measurements to the force. The strain ϵ is the amount of deformation a material experiences due to an applied force

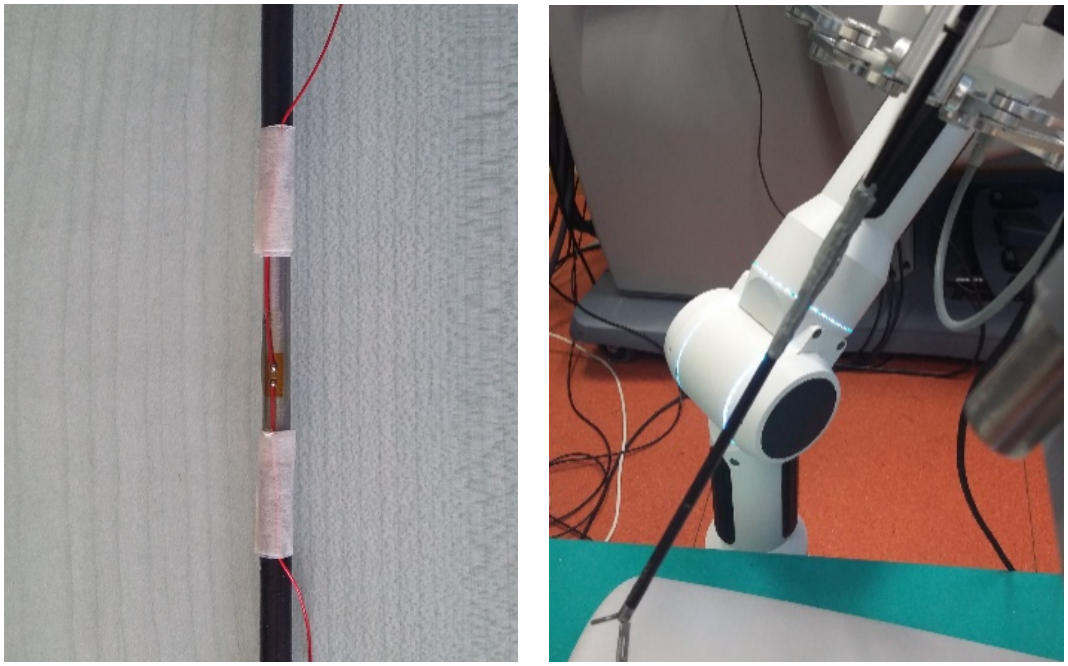
$$\epsilon = \frac{\Delta L}{L}$$

where ΔL is the ratio of the change in length of a material to the original unaffected length L. A strain gauge is a sensor whose resistance varies with

applied force. It converts a change in electrical resistance into a force which can then be measured, and can be used to pick up expansion as well as contraction. The metallic strain gauge consists of a very fine wire or, as in our case, metallic foil arranged in a grid pattern. The grid pattern maximizes the amount of metallic wire subject to strain in the parallel direction. The grid is bonded to a thin backing called the carrier, which is attached directly to the body whose bending is to be measured. Therefore, the strain experienced by the body is transferred directly to the strain gauge, which responds with a linear change in electrical resistance. The Gauge Factor G_f is the strain gauge sensitivity to strain, and it is the ratio of the fractional change in electrical resistance $\frac{\Delta R}{R}$ to the fractional change in length ϵ

$$G_f = \frac{\Delta R}{R\epsilon}.$$

In practice, since strain measurements rarely involve quantities larger than few milli-strain, strain gauges can accurately measure only very small variation in resistance (i.e., force). To measure such small changes in resistance, strain gauge relies on a Wheatstone bridge. As shown in Figure 2.5, since we are interested in measuring the forces applied on the laparoscopic tool in xyz-directional axes, three strain gauges are mounted arranged 120 degrees apart on the laparoscopic tool's shaft. A custom board, equipped with a Nucleo-STM32L432KC, has been developed to measure the voltage signal from the strain gauges and to perform the analogue to digital conversion. The board has six Wheatstone bridges which can be arranged in a quarter-bridge or half-bridge configuration and each bridge has an independent power supply regulator to minimize the disturbances between them. When the board is configured as a quarter bridge, as in our case, the board uses a high precision resistor (0.01%) and high thermal stability (0.2 ppm/$^\circ C$). The Wheatstone bridge was design to read up to ± 1000.0 micro-strain (in a quarter bridge) which maps to a range of $\pm 7.0N$ of force. However, strain gauges arranged as in our configuration cannot measure compression along the main axis of the laparoscopic tool (i.e., z-axis). This is because the tool's deformation along such axis is lower than an order of magnitude with respect to the other axes. For this reason, the interaction force along the z-axis is not estimated and neither shown in the plots. Finally, the static deformations of the tool, the contribution of the gravity and the vibrations introduced by the motion of the end-effector are threshold.



(a) (b)

Figure 2.5: Mouting of the strain gauges on the grasper tool. a) a single strain gauge applied on the laparoscopic tool before being covered with protective material, b) three strain gauges applied on the *SARAS* laparoscopic tool after being covered with the protective material.

Strain gauges calibration

The calibration procedure aims to identify the matrices $\boldsymbol{A}_1$, $\boldsymbol{A}_2$ and the vector $\boldsymbol{g}$ in the linear systems

$$\begin{cases} \boldsymbol{f} = \boldsymbol{A}_1 \boldsymbol{v} + \boldsymbol{g} \\ \boldsymbol{\Delta x} = \boldsymbol{A}_2 \boldsymbol{v} \end{cases} \tag{2.6}$$

where $\boldsymbol{f} \in \mathbb{R}^3$ is the force along the tree axes (since we are mapping only translation forces), $\boldsymbol{v} \in \mathbb{R}^3$ is the measured voltage on the three strain gauges, $\boldsymbol{\Delta x} \in \mathbb{R}^3$ is the Cartesian displacement between the measured and the real position of the tool, $\boldsymbol{g} \in \mathbb{R}^3$ is the gravity, and $\boldsymbol{A}_i \in \mathbb{R}^{3\times 3}$ with $i \in \{1, 2\}$ are the matrices mapping voltages into forces and and tool-displacement, respectively. We define $t = k\Delta t$ as the current time and Δt the sample time of the sensing system. Force samples $\boldsymbol{f}(k)$ for the calibration are provided by a 3-DOF ATI Micro force sensor. The voltage samples $\boldsymbol{v}(k)$ are obtained directly by the strain gauges acquisition board. Let $\boldsymbol{x}_f$ be the fixed position of the force sensor on which the tool tip is in contact, and $\boldsymbol{x}(k)$ be the position of the laparoscopic tool tip estimated by the kinematics. The displacement between the tip poses of the real *flexible* tool and the *assumed rigid* tool is $\Delta\boldsymbol{x}(k) = \boldsymbol{x}_f - \boldsymbol{x}(k)$.

The samples $\boldsymbol{f}(k)$ are collected by performing a sequence of motion in

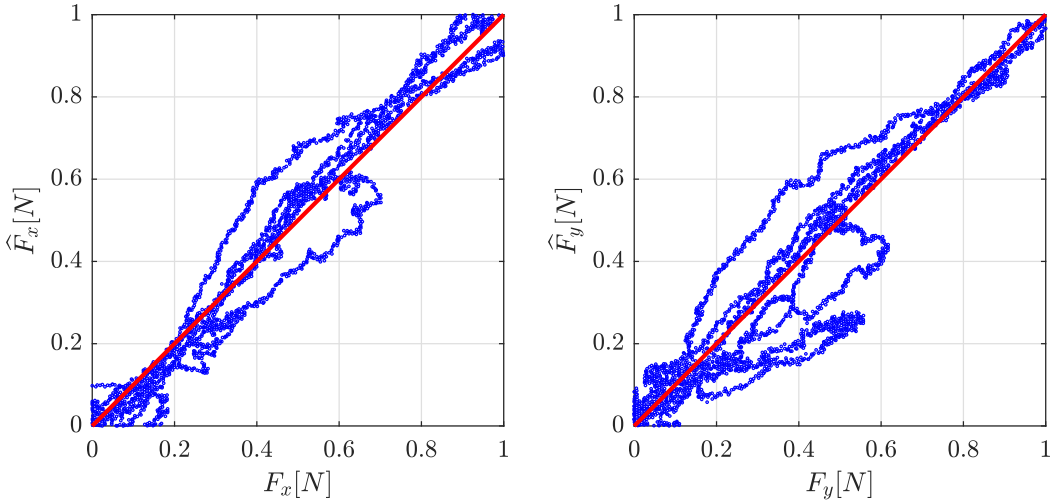


Figure 2.6: Linear ratio between measured and computed forces on the x and y axes.

which we touch the force sensor with the laparoscopic tool tip in different configurations and in free-motion to estimate the gravity contribute. At each time step k, the samples $\boldsymbol{v}(k)$ and $\boldsymbol{f}(k)$ are added to the samples matrices $\boldsymbol{V}_1(k)$ and $\boldsymbol{F}(k)$

$$\boldsymbol{V}_1(k) = \begin{bmatrix} 1 & \boldsymbol{V}_1(k-1) \\ 1 & \boldsymbol{v}(k)^T \end{bmatrix}, \ \boldsymbol{F}(k) = \begin{bmatrix} \boldsymbol{F}(k-1) \\ \boldsymbol{f}(k)^T \end{bmatrix}, \tag{2.7}$$

and the samples $\boldsymbol{v}(k)$ and $\Delta\boldsymbol{x}(k)$ are added to the samples matrices $\boldsymbol{V}_2(k)$ and $\Delta\boldsymbol{X}(k)$

$$\boldsymbol{V}_2(k) = \begin{bmatrix} \boldsymbol{V}_2(k-1) \\ \boldsymbol{v}(k)^T \end{bmatrix}, \ \Delta\boldsymbol{X}(k) = \begin{bmatrix} \Delta\boldsymbol{X}(k-1) \\ \boldsymbol{x}(k)^T \end{bmatrix}, \tag{2.8}$$

if the laparoscopic tool is in contact with the force sensor. The final matrices are $\boldsymbol{V}_1 \in \mathbb{R}^{3\times n}$, $\boldsymbol{F} \in \mathbb{R}^{3\times n}$ and $\boldsymbol{V}_2 \in \mathbb{R}^{3\times m}$, $\Delta\boldsymbol{X} \in \mathbb{R}^{3\times m}$, where n is the total number of samples and m is the number of collected samples in contact with the force sensor. The matrices $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ are then computed with the least squares method as

$$\Big[\boldsymbol{A}_1 | \boldsymbol{g}\Big] = (\boldsymbol{V}_1\boldsymbol{V}_1^T)^{-1}(\boldsymbol{V}_1\boldsymbol{F}) \tag{2.9}$$

$$\boldsymbol{A}_2 = (\boldsymbol{V}_2\boldsymbol{V}_2^T)^{-1}(\boldsymbol{V}_2\Delta\boldsymbol{X}). \tag{2.10}$$

Using the same method we also compute the regression matrix which related voltages to tool bending. Figure 2.6 shows the proportional relationship between the force measured by an ATI-Mini force sensor and the strain gauges equivalent force.

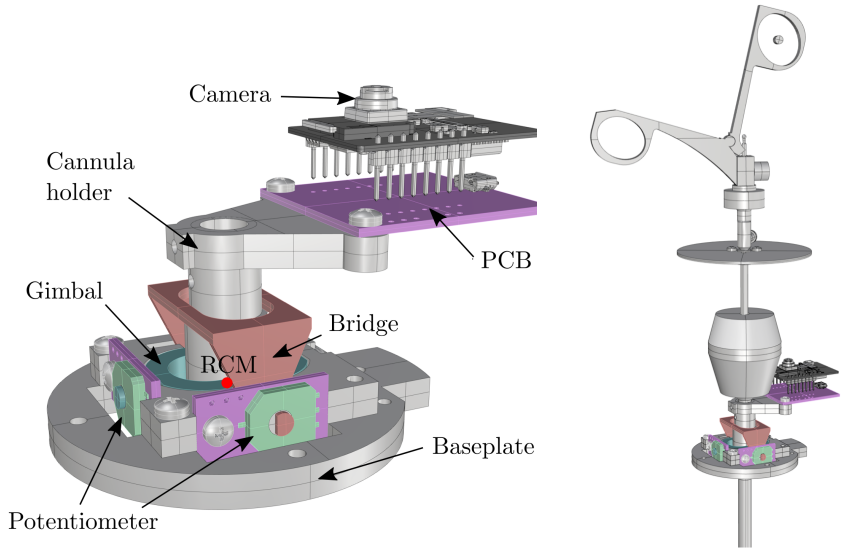
2.3 Smart trocar

In the LAPARO2.0-SURGERY platform the main surgeon performs the surgery using a set of standard laparoscopic instruments. In such situation the position and the orientation of the laparoscopic tools' tip cannot be retrieved directly. For this reason we designed and manufactured a pair of *smart trocars*: a trocar holder able to estimate the position and the orientation of a trocar, and so of the laparoscopic tool inserted in it. To have reliable measurements of the instrument tip's pose we also designed a holding system for the smart trocars which should be placed in contact with the belly of the patient.

The design of the smart trocar, shown in Figure 2.7, integrates a sensorized platform with standard laparoscopic trocars. The tool diameter considered in this design is 5 mm, but it can be easily adapted to any trocar size. It is built around a gimbal and bridge component that transfers the pivoting motion of the trocar cannula holder to two potentiometers. A PCB is attached to the side of the cannula to host a microcontroller and a camera. This mounting position allows the camera to track a set of markers painted on a disk on the laparoscopic instrument which does not interfere with the trocar's motion as shown in Figure 2.8. The camera is used to measure the rotation and insertion of the tool. Assuming a mounting position in contact with the patient epidermis, the effective Remote Center of Motion (RCM) is 9 mm above. This is not a major concern as the maximum radial deformation at the insertion point is ≈ 10 mm given the maximum longitudinal and transversal rotation of the gimbal and bridge of 40° and 64°, respectively. For this reason, the forces acting on the epidermis are acceptable.

The microcontroller (an ESP32-CAM model) handles the raw measurements from the potentiometers via the PCB and synchronizes them with the camera frame acquisition. It communicates this information through WiFi via the *rosserial* protocol to a computer that estimates the pose of the laparoscopic tool tip (Tool Center Point, TCP). The proposed method computes the position and orientation frame of the TCP $\boldsymbol{T}_e^r$ with respect to the mechanical RCM placed in the middle of the smart trocar base plate. Let $\boldsymbol{T}_r^w$ be the transformation of the RCM with respect to a global reference frame; in our use-case we attached the smart trocar to a rigid stand as shown in Figure 2.9 and the transformation is estimated by using the methodology that will be discussed in Chapter 3. $\boldsymbol{T}_e^r$ can be written as

$$\boldsymbol{T}_e^r = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{pmatrix} \tag{2.11}$$



(a) Smart trocar (b) Full assembly

Figure 2.7: CAD of the smart trocar prototype. a) a close view of the base plate, b) a complete view with the tool and trocar mounted on it.

where $\boldsymbol{R} = \boldsymbol{R}(\psi)\boldsymbol{R}(\theta)\boldsymbol{R}(\phi)$ and $\boldsymbol{t} = \begin{bmatrix} 0 & 0 & t_z \end{bmatrix}$. The position of the TCP $\boldsymbol{T}_e^w$ with respect to the global reference frame is then

$$\boldsymbol{T}_e^w = \boldsymbol{T}_r^w \boldsymbol{T}_e^r. \tag{2.12}$$

The Euler angles ψ and θ are measured thanks to two potentiometers mounted on the board while ϕ is the instrument rotation and is estimated using the camera and the markers. We aligned the rotation axis of the sensors with a set of Cartesian axes having their z-axes aligned with the instrument shaft. To measure both the distance and the rotation ϕ we used a set of eight ArUcO markers [25] placed on a support disk anchored to the laparoscopic tool as shown in Figure 2.8. The support has been designed in such a way to always have at least one of these markers visible by the camera mounted on the support. The ArUcO marker is a fiducial marker system specially designed for camera pose estimation with a configurable set of marker dictionaries (in size and number of bits). The marker, as shown in Figure 2.8 is a synthetic square composed by a wide black border and an inner binary matrix which determines its identifier. The binary codification allows its identification and the application of error detection and correction techniques [25].

The distance is estimated by assuming that the camera and the disk remain parallel to each other which allows to average the depth of all identified ArUcO markers. Finally, the rotation ϕ is estimated by measuring the rotation angle

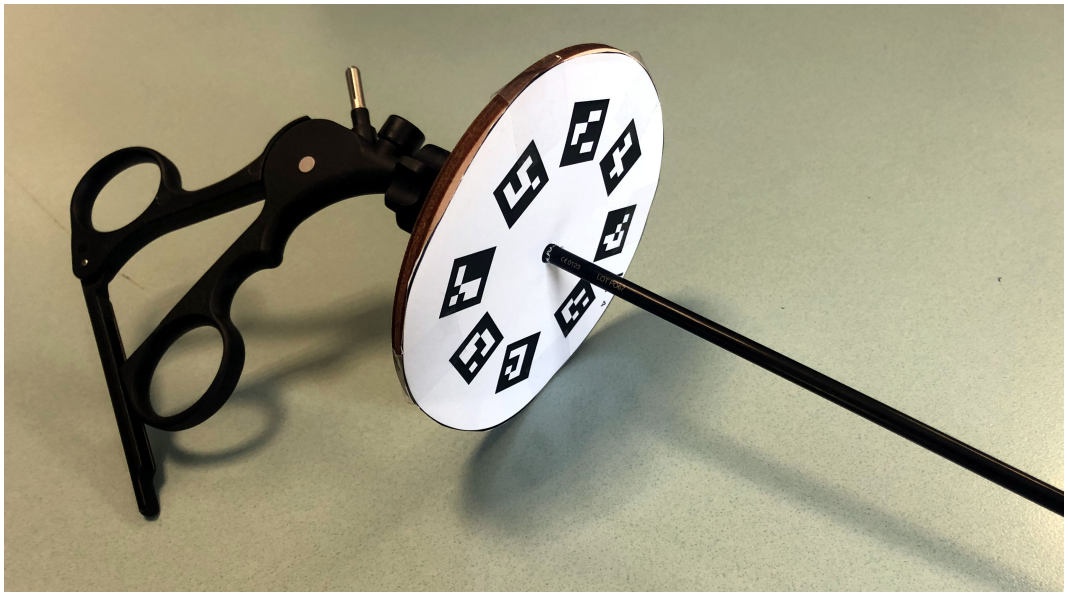


Figure 2.8: The smart trocar's disk used to estimate the insertion and rotation values of the surgical instrument.

of each marker on the face-plate plane. The ArUcO-based insertion measurements were found to be more precise and reliable than the early design that used a *time of flight* (ToF) sensor, with the added bonus of being able to also estimate the tool's rotation along its main axis. The proposed smart trocar prototype, as shown in Figure 2.9, has been tested on an experimental setup with one of the Panda arm acting as the ground truth position reference and a Storz® grasper laparoscopic tool as the surgeon instrument. We registered the robot and the smart trocar prototype in a shared reference frame to later compare their measurements. We performed a sequence of motion towards a fixed position in an anatomical manikin having an impedance similar to the human skin at the RCM. Then, we recorded the target positions using the robot kinematics and, as shown in Figure 2.10, we compared them with the measured positions of the laparoscopic TCP over multiple tests with a maximum error of 4 mm. Such error is deemed sufficient for the intended application. Such error can be mostly attributed to the mechanical play related to the materials used in the prototype.

2.4 Anatomical manikins

To evaluate the semi-autonomous system in the two case studies, two different anatomical manikins (also called phantoms) have been used: a model of the lower abdominal for the RARP and a model of the left upper abdominal

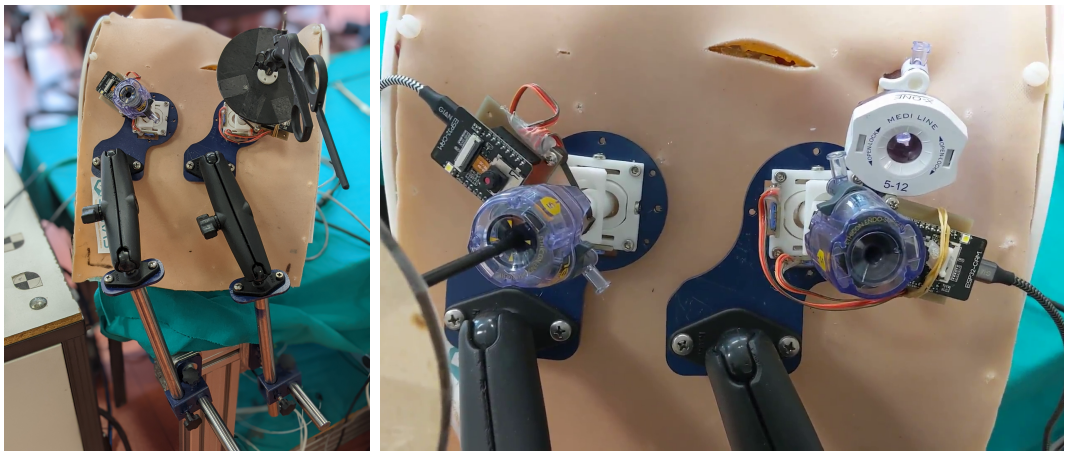


Figure 2.9: The 3D printed final prototype of the smart trocar used during the experimental validation; the trocars are mounted on a pair of adjustable arms to adapt to the anatomy; the diameter of the smart-trocar cannula is 5 mm.

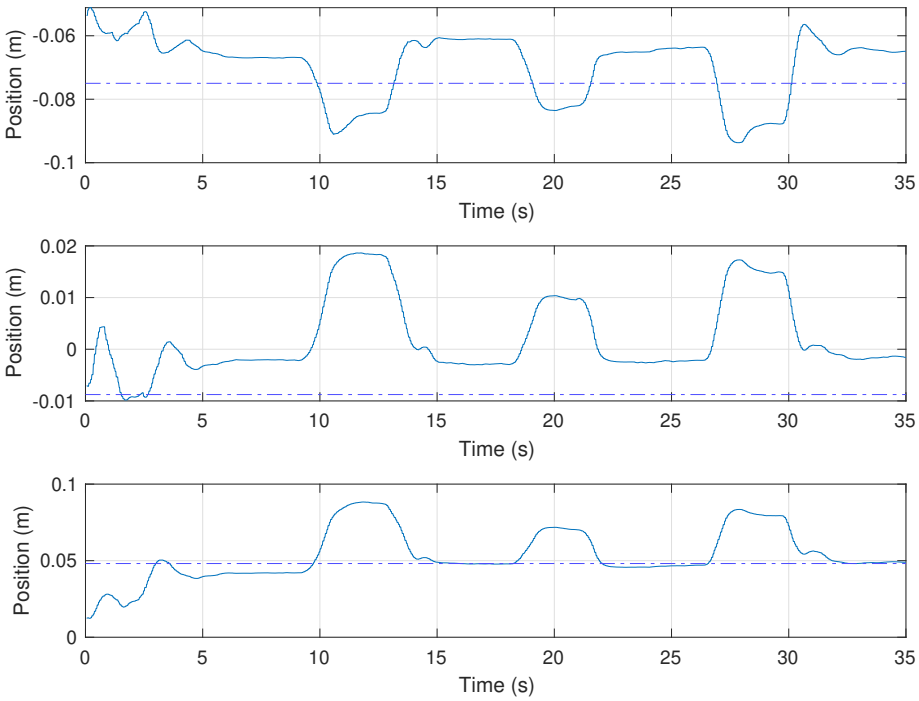
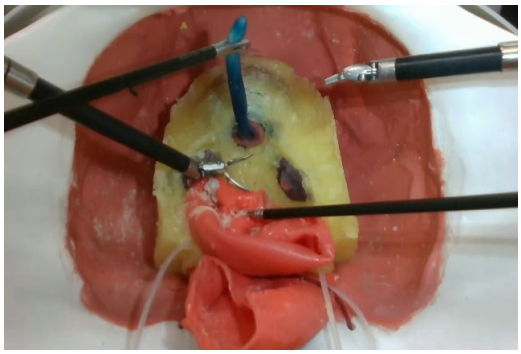


Figure 2.10: The experimental validation of the proposed prototype: the solid line shows the laparoscopic tool position and the dashed line shows the reference position of the landmarks.

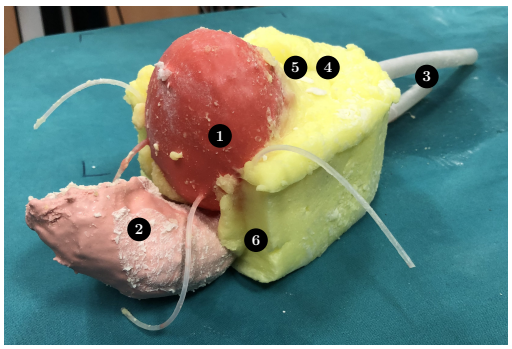
quadrant for the LRN.

Lower abdominal model. The phantom, as shown in Figure 2.11b and Figure 2.11a, is composed of a 3D-printed pelvis bone and the following silicone elements: rectum, bladder, urethra, seminal vesicles, vas deferens, and prostate, all immersed in a silicate fat-like foam. The phantom has been developed by ACMIT *Gmbh*, Austria, a partner of the *SARAS* project consortium.

Left upper abdominal quadrant model. The phantom, as shown in Figure 2.11c, is composed of a 3D-printed abdominal walls containing a model of the kidney embedded into a foam-like silicate representing the Gerota. The internal model, shown in Figure 2.11d, is composed of the kidney organ, the renal artery, the renal and gonadal veins and the ureter. The phantom is developed by the University of Dundee, partner of the *SARAS* project consortium.

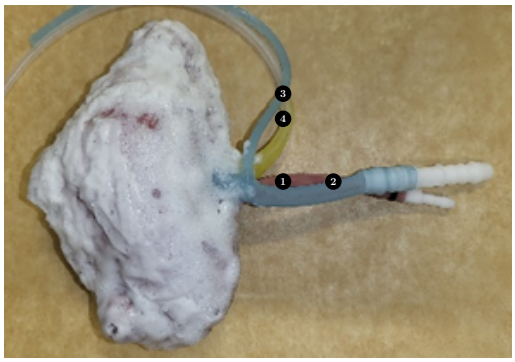


(a) RARP external view



(b) Prostate manikin

(c) LRN External view



(d) Kidney manikin

Figure 2.11: The anatomical manikins used during the evaluation of the RARP and the LRN. a) the da Vinci® robotic tools, the *SARAS* robotic tools and the phantom used during an experimental validation, b) the phantom is composed of bladder (1), rectum (2), urethra (3), prostate (4), seminal vesicles (5), fat (6), c) the external view of the abdominal phantom, d) the kidney model comprises the renal artery (1), renal vein (2), gonadal vein (3) and the ureter (4).

Chapter 3

Registration and calibration

This chapter deals with the preliminary technologies needed by a semi-autonomous robotic system to operate. The first requirement is the capability of creating a common reference frame shared between multiple robotic arms and external sensors, like cameras. While the second requirement is the capability of projecting pre-operative data into the shared working environment. In a multi-arm robotic setup the creation of a shared reference frame allows to perform complex manipulation. For instance, dual arm manipulation or collision avoidance needs to have a precise relative positioning of all the arms and the obstacles in the environment. A common way to reconstruct a 3D model of the surrounding environment is to use vision systems. In the last decade, mono and RGB-D stereo cameras played an important for real-time 3D reconstruction. Once the common reference system has been established and a 3D reconstruction of the environment is available, registering pre-operative data in the environment allows to project plans and objects which can be later used by the semi-autonomous system to perform the required tasks.

3.1 State of the art

Better medical imaging and vision techniques have significantly improved the performance of robotic surgical systems in a range of clinical scenarios, such as orthopaedics and neurosurgery [28]. Vision systems can retrieve pre and

Part of this chapter is based on the following publications:

- A. Roberti, N. Piccinelli, D. Meli, R. Muradore, and P. Fiorini, "Improving rigid 3-d calibration for robotic surgery," *IEEE Transactions on Medical Robotics and Bionics*, vol. 2, no. 4, pp. 569–573, 2020
- N. Piccinelli, A. Roberti, E. Tagliabue, F. Setti, G. Kronreif, R. Muradore, and P. Fiorini, "Rigid 3d registration of pre-operative information for semi-autonomous surgery," in *2020 International Symposium on Medical Robotics (ISMR)*, 2020, pp. 139–145. DOI: 10.1109/ISMR48331.2020.9312949

intra-operative information from tomography (CT), magnetic resonance (MR) and ultrasound to plan tool trajectories and support surgeons' decision making. However, image-guided interventions require an accurate calibration to map poses of robots, instruments and anatomy into a common reference frame. Hand-eye calibration has been widely studied within the robotics literature [29]. The Hand-eye calibration problem (also called the robot-sensor or robot-world calibration problem) is the problem of determining the transformation between the robot end-effector and a set of sensors (like cameras or laser scanners) or between the robot base and the world coordinate system [30].

In the surgical domain all these challenges are even more complicated to solve due to the complexity of the anatomical environment. In general, a surgical intervention is planned using pre-operative information about the patient and then the surgeons use this knowledge to decide which actions to take. In autonomous systems, the mapping of this planning to the patient anatomy is not trivial since it has to be updated in real-time during the intervention, based on the actual patient condition. This scenario is further complicated by the fact that the environment is soft, which means that it could be subjected to deformation and some anatomical structures can be removed during surgery. Perception of the current surgical environment during R-MIS is usually performed with a stereo endoscope that provides visual feedback to the main surgeon. Although it is able to produce a 3D dense reconstruction of the environment, the main drawback of a stereo vision approach is that the endoscope has to be sufficiently close to the region of interest to provide a reliable reconstruction. Monocular vision systems on the other hand cannot reconstruct a dense point cloud in real-time, but exploiting the multi-view approach, they can provide sparse and dense 3D reconstruction. In order to apply the multi-view approach the camera must be moved over time and the scene should be almost static.

In R-MIS systems, where the patient-side arms are constrained by a RCM, it is challenging to obtain the camera motion range needed to guarantee an accurate calibration from the numerical point of view. Wang [31] takes advantage of this constraint by finding a unique relationship between the endoscope and the surgical tool using camera perspective projection geometry. A different approach is followed in [32], [33] where the instruments themselves are used as calibration tools. Thus far, several closed-form solutions for 2D images have been proposed for hand-eye calibration that use linear methods that separate rotations and translations. In [34], the orientation component was derived by utilizing the angle-axis formulation of rotation, then the translational component was estimated using standard linear regression techniques. Chou and

Kamel [35] introduced quaternions to represent orientation and estimated the quaternion coefficients by solving a homogeneous linear least squares problem. A closed form solution was then derived using the generalized inverse method with singular value decomposition analysis. Other works [36]–[38] used the Kronecker product to get a homogeneous linear equation for the rotation matrix. However, such solutions separate the rotational and translational components neglecting the intrinsic correlation between them, resulting in poor calibration accuracy.

In [39] the authors studied the comparison between hand-eye calibration based on 2D and 3D images, introducing quantitative 2D and 3D error metrics to assess the calibration accuracy. They proved that the 3D calibration approach provides more accurate results on average but requires burdensome manual preparation and much more computation time than 2D approaches. Kim used 3D measurements at the center of markers for the hand-eye calibration [40]. Fuchs [41] proposed a solution based on depth measurements instead of 2D images, using a calibration plane with known position and orientation. The hand-eye calibration was then obtained by estimating the best fitting plane of the measured depth values.

Simultaneous localisation and mapping (SLAM) is a technique applied to obtain the 3D structure of an unknown environment and to estimate the camera motion. SLAM was originally proposed to design autonomous control architecture for mobile robots and then spread to other fields like computer vision-based online 3D modelling, augmented reality (AR)-based visualisation, and self-driving cars. In recent years, many efforts have been made to assess the feasibility of applying SLAM in laparoscopy to reconstruct a sparse or even dense soft-tissue surface [42]–[45]. A very popular approach for SLAM in minimally invasive surgery (MIS) relies on oriented FAST and rotated BRIEF (ORB) features [46]. Several works have proved that this approach can successfully support the process of endoscope localisation by providing the poses which are necessary to create a quasi-dense map of the environment [47], [48]. When a monocular vision system is used, there is an additional challenge based on the scale factor estimation. In [49], for example, they retrieve the scale factor manually through the computed tomography (CT) scan and apply it to the 3D reconstruction before the registration.

Once the partial intra-operatively reconstruction is available, its correct registration with the pre-operative information is a fundamental capability to enable the robot to perform tasks like object detection and recognition, navigation and 3D dense map reconstruction. Point cloud registration aims to estimate the homogenous transformation matrix between two point cloud

scans. Applying such a matrix we can merge partial scans of the same 3D scene or object into a complete 3D point cloud. Point cloud registration has many applications in critical aspects of computer vision, in first 3D reconstruction. Generating an entire 3D scene is an essential and significant technique for various computer vision applications, including high-precision 3D map reconstruction in autonomous driving, 3D environment reconstruction in robotics and 3D reconstruction for real-time monitoring and decision-making. Another crucial registration application is 3D localization, used to compute the position of an agent in a 3D environment. Point cloud registration could accurately match a current real-time 3D view to its belonging 3D environment to provide a high-precision localization service. Finally, pose estimation. In this case, aligning point cloud A (3D real-time view) to another point cloud B (the 3D environment) could generate the pose information of point cloud A related to point cloud B.

In the context of autonomous robotic surgery, this represents an essential step to plan the robot motion. For example, in [50], [51] a CT scan is used to select the ablation points and the extracted model is then registered to the phantom using embedded spherical landmarks. However, from a practical point of view, the use of landmarks is rarely feasible within a realistic surgical environment. The most popular approach that allows to rigidly align models when no information about landmarks or correspondences is available is the Iterative Closest Point (ICP). ICP solves the registration by iteratively computing a transformation by reducing a distance metric between an intra-operative 3D surface scan and a preoperative 3D model. As opposed to other registration algorithm, ICP treats correspondences between the two scans as a variable to be estimated. The main drawback of this method is that the standard implementation does not always guarantee to find the globally optimal transformation because it can be easily trapped in local minima [52]. Several advanced implementations of ICP have been proposed to tackle this issue, which either rely on the extraction of robust features [53] or exploit more efficient ways to search the 3D space [54], [55].

3.2 Multi arm 3D calibration

The multi-arm calibration method proposed in this thesis uses an RGB-D camera and differently from [41], the accuracy and computational time do not depend on the placement of the calibration board within the workspace. The approach is based on the idea of subdivide the calibration procedure in two

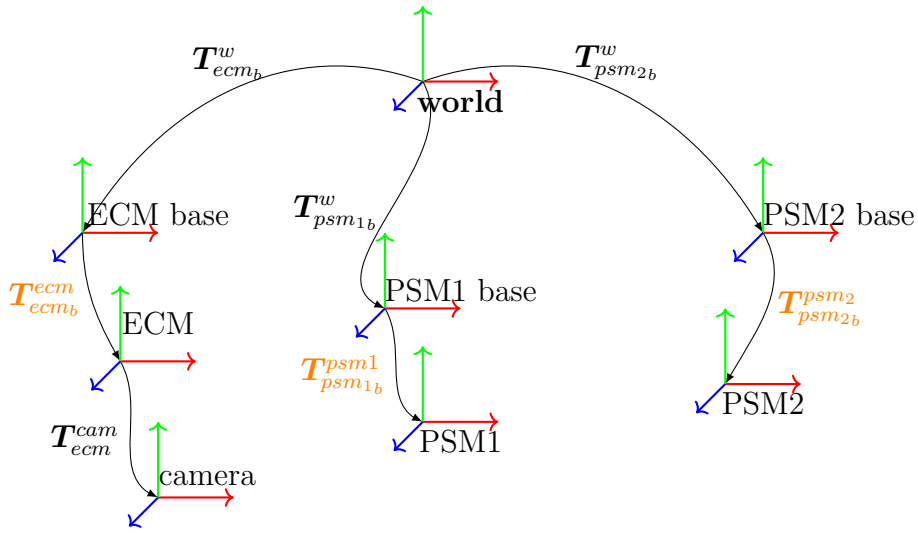


Figure 3.1: The reference frames produced by our proposed method (the axes direction of the reference frames are only for visualisation purpose). The orange transformations are known, whereas the black transformations are to be estimated.

main parts. We initially proceeds with the calibration of the robotic arms and then we perform the hand-eye calibration of the camera. Each part implements a three-step method with a closed-form solution:

1. **Arm calibration** performed by touching reference points with the end-effectors of the surgical robot on a custom calibration board

2. **Camera calibration** performed by recognizing the same reference points with the RGB-D camera

3. **Hand-eye calibration** performed by mapping the poses reached by the robotic arms in the first step to the 3D points computed in the second step.

The main advantage of the proposed method is the improved accuracy in a 3D metric space, which is increased by a factor of four with respect to the state-of-the-art results with comparable sensors [39]. In our case study, in the first part we will estimate the homogenous transformations $\boldsymbol{T}^{w}_{\star}$ between the common reference frame ($world$) and the base frames of the da Vinci® arms $\star \in \{ecm_b, psm_{1b}, psm_{2b}\}$. Finally, the hand-eye transformation $\boldsymbol{T}^{cam}_{ecm}$ between the camera reference frame and the ECM reference frame is estimated. The resulting transformation tree is shown in Figure 3.1. The procedure starts by positioning the calibration board in the robot workspace. We choose a set P of reference points such that each point $\boldsymbol{p} \in P$ is reachable by the three arms and visible from the camera. The points in P must be symmetric with respect

to the center of the board to compute the origin of the common reference frame; at least three points are needed to estimate the plane coefficients. The best fitting plane is characterized by the centroid $\boldsymbol{c}$ of the point set P, and its normal vector $\boldsymbol{n}$. Their optimal estimations are the solution of the optimisation problem

$$\{\hat{\boldsymbol{c}}, \hat{\boldsymbol{n}}\} = \arg\min_{\boldsymbol{c},\boldsymbol{n}} \sum_{i=1}^{n} ((\boldsymbol{p}_i - \boldsymbol{c})^T \boldsymbol{n})^2 \tag{3.1}$$

$$\text{subject to} \|\boldsymbol{n}\|_2 = 1 \tag{3.2}$$

As in [56] the centroid is estimated as

$$\hat{\boldsymbol{c}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{p}_i. \tag{3.3}$$

The normal vector $\hat{\boldsymbol{n}}$ is obtained by factorizing the distance matrix $\boldsymbol{A}$ with the Singular Value Decomposition (SVD)

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T \tag{3.4}$$

where $\boldsymbol{A} : K^n \rightarrow K^m$. SVD is a factorization technique which generalizes the eigendecomposition of a square matrix to any matrix shape. One of the main applications of such factorization is for the computation of the pseudo-inverse of a matrix which is one of the ways of solving linear least squares problems. Since the matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ are unitary, their column vectors yield to an orthonormal basis for K^n and K^m respectively. Starting from the decomposition

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{p}_1 - \hat{\boldsymbol{c}}, \ldots . \boldsymbol{p}_n - \hat{\boldsymbol{c}} \end{bmatrix} \tag{3.5}$$

the normal vector can be obtained by taking the third column of the matrix $\boldsymbol{U} = \begin{bmatrix} \boldsymbol{u}_1 & \boldsymbol{u}_2 & \boldsymbol{u}_3 \end{bmatrix}$, i.e., $\hat{\boldsymbol{n}} = \boldsymbol{u}_3$.

Arm calibration

To find the transformation of the arms base frame with respect to the common reference frame we record the end-effector pose of the arms (PSMs and ECM with adapter) on each point in the set P. In order to obtain the ECM effective pose, we remove the known rigid transformation between the adapter and the ECM. On this set we estimate the best fitting plane using (3.2). The set P is then augmented by adding a point above the calibration board acquired by moving the arm's end-effector. This last point is used to define the desired plane normal direction

$$\boldsymbol{n}_d = \frac{\boldsymbol{p}_{n+1} - \hat{\boldsymbol{c}}}{|\boldsymbol{p}_{n+1} - \hat{\boldsymbol{c}}|_2}$$

where $\boldsymbol{p}_{n+1}$ is the last point in the ordered set P, $\hat{\boldsymbol{c}}$ is the centroid of P and $|*|_2$ is the vector norm. For each arm, the homogeneous transformation $\boldsymbol{T}_\star^w$ of the common reference with respect to the arm base frame is defined using the unit vectors

$$\begin{aligned}
\boldsymbol{u} &= \text{sign}(\boldsymbol{n} \cdot \boldsymbol{n_d})\boldsymbol{n} \\
\boldsymbol{l} &= \boldsymbol{u} \times \frac{\boldsymbol{p}_1 - \hat{\boldsymbol{c}}}{|\boldsymbol{p}_1 - \hat{\boldsymbol{c}}|_2} \\
\boldsymbol{f} &= \boldsymbol{l} \times \boldsymbol{u}
\end{aligned}$$

and the centroid $\hat{\boldsymbol{c}}$,

$$\boldsymbol{T}_\star^w = \begin{bmatrix} \boldsymbol{f}_x & \boldsymbol{l}_x & \boldsymbol{u}_x & \hat{\boldsymbol{c}}_x \\ \boldsymbol{f}_y & \boldsymbol{l}_y & \boldsymbol{u}_y & \hat{\boldsymbol{c}}_y \\ \boldsymbol{f}_z & \boldsymbol{l}_z & \boldsymbol{u}_z & \hat{\boldsymbol{c}}_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The use of $\boldsymbol{p}_1$ is just to have the x direction of the reference frame pointing to the first calibration point on the calibration board.

Camera calibration

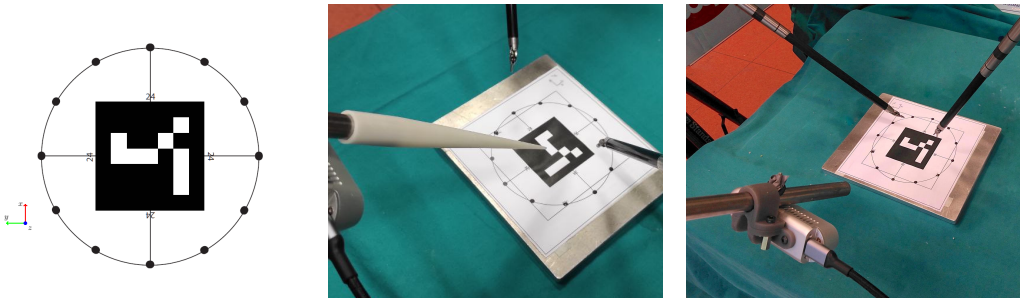
To find the transformation $\boldsymbol{T}_{cam}^w$ for the RGB-D camera we first detect the center of the ArUco marker on the board with respect to the camera frame. Once we find a camera position that ensures good visibility and a stable pose of the ArUco marker, we align the pose on the point cloud generated from the depth map acquired by the RGB-D camera. We use the marker pose and its known radius to generate the pose of every dot, as shown in Figure 3.2a, in the set P in the marker reference frame, as well as the point above the calibration board. Once the poses of each $p \in P$ is obtained, we find the best fitting plane using (3.2) and then we build the homogeneous transformation $\boldsymbol{T}_{cam}^w$ between the common reference frame to the camera base frame by adapting the previous approach used for the arms.

Hand-eye calibration

The hand-eye calibration problem is formulated using the homogeneous transformation matrices:

$$\boldsymbol{AX} = \boldsymbol{XB}$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ are known homogeneous matrices representing the frames of the base of the robot and the camera, respectively. The unknown transformation $\boldsymbol{X}$ is between the robot coordinate frame and the camera coordinate



(a) Calibration pattern (b) ECM adapter (c) Setup

Figure 3.2: The calibration components and the experimental setup. a) the calibration board with the marker, the coloured axes represents the common reference frame directions b) the adapter for the ECM positioning, c) the proposed setup for calibration, with the RealSense d435, the PSMs and the calibration pattern.

frame. Given $\boldsymbol{T}_{cam}^{w}$, we can compute $\boldsymbol{X}$ as the relative homogeneous transformation between the end-effector of the ECM and the RGB-D base frame

$$\boldsymbol{T}_{ecm}^{cam} = \boldsymbol{T}_{w}^{cam}(\boldsymbol{T}_{w}^{ecm})^{-1}.$$

3.2.1 Calibration accuracy evaluation

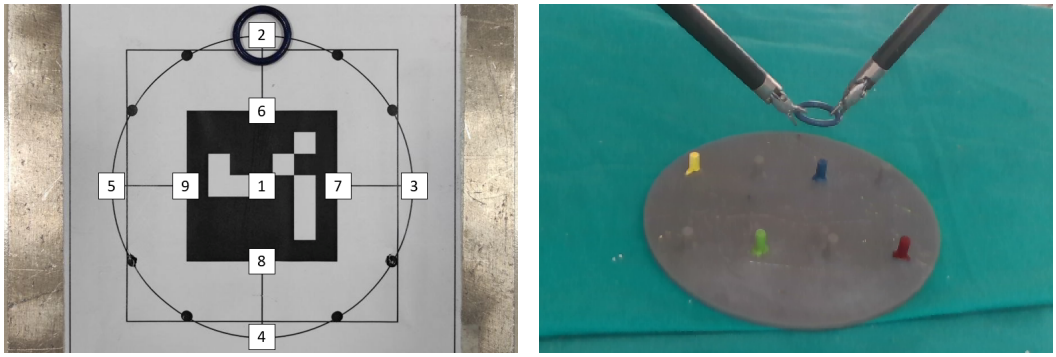
The validation of the proposed method has been carried out with the dVRK robot adopting the custom calibration board shown in Figure 3.2a. The ArUco marker is placed in the center of a circle of 50 mm radius surrounded by several reference dots. We have also equipped the ECM with a 3D-printed adapter, shown in Figure 3.2b, to have a thinner end-effector to guarantee precise positioning on the dots on the board. The whole calibration method has been implemented in ROS using the Point Cloud Library (PCL) and OpenCV. We compared our calibration with the Tsai's method [57] in two benchmark tests for surgical robotics:

- Localization and grasping of small targets,
- Dual-arm manipulation

Finally we evaluated the accuracy of the projection from 2D camera image plane to the 3D workspace.

Localization and grasping

In the first scenario (Figure 3.3a) the two PSMs must autonomously grasp a ring placed on the calibration board, in this case on location 2. The RGB-D



(a) Localization and grasp (b) Dual-arm manipulation

Figure 3.3: Experimental setup for the calibration accuracy evaluation. a) setup for the localization and grasping experiment, the numbers on calibration board represents the nine grasping locations, b) the dual arm manipulation experiment consists in two arms carrying a ring while performing circular trajectories through the workspace.

camera identifies the point cloud corresponding to the ring after color and shape segmentation, and points are transformed from the camera to the common reference frame. The ring has a diameter of 15 mm, and the target point for both PSMs is chosen as the center of the ring. The ring is placed in the 9 different locations on the board to cover the full $x - y$ plane, as shown in Figure 3.3a. The arms reach the target points ten times, and for each iteration we compute the Euclidean distance between the target and the final positions of the PSMs. In this way, we estimate the mean accuracy of our calibration procedure on the $x - y$ plane. The results are reported in Figure 3.4 and compared with state-of-the-art Tsai's calibration method [57]. It is worth mentioning that errors are comprehensive of the estimated kinematic accuracy of the da Vinci®: 1.02 mm on average when localizing and reaching fiducial markers [58], with a maximum error of 2.72 mm [59]. Table 3.1 shows that our

Table 3.1: A comparison of the error in the localization and grasping test

	Max error (mm)	Mean error (mm)	Std dev (mm)
Our method	1.07	0.53	0.15
Tsai [57]	3.17	1.83	0.33

method achieves significantly better accuracy (0.53 mm average error against 1.83 mm with Tsai's calibration). The error does not depend on the location of the ring on the $x - y$ plane.

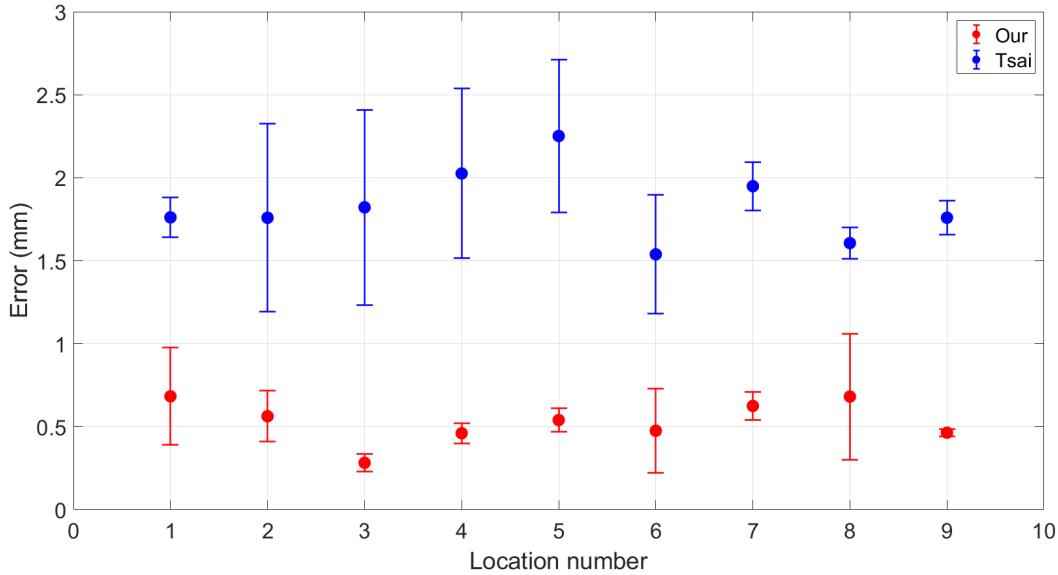


Figure 3.4: The measured 3D positioning errors between the end-effector and the grasping point during the localization and grasping experiment. The error bars shows the mean and standard deviation across the nine grasping locations.

Dual arm manipulation

In the second scenario (Figure 3.3b) the PSMs start holding the same ring, and execute simultaneous pre-computed circular trajectories with centre on the z axis of the common reference frame (45 mm above the calibration board) and radius r ranging from 10 mm to 40 mm. Circumferences are first defined in the $x - z$ plane of the common reference frame (normal to the calibration board), and then replicated in planes rotated around the z axis with a step of 10 deg. In this way we define a spherical workspace by interpolation between the recorded trajectories. PSMs are commanded with the transformed waypoints in their relative frames. This task validates the accuracy of the transformations between the arms computed with the proposed method. We measure the difference between the trajectories of the two PSMs, and we consider the standard and the maximum deviations from the mean for each radius. With no calibration and kinematic errors, the difference between the trajectories should have null standard deviation. Figure 3.5 shows the absolute error through the workspace for spheres with radii 20 mm, 30 mm and 40 mm, by using the Lambert equal-area cylindrical projection [60]. In Table 3.2 we report the errors for all the spheres. We notice that the mean error increases with the radius of the sphere, as the PSMs move away from the calibration plane. The standard deviation of the error increases with the radius but remains below 0.11 mm, hence the overall error does not change significantly on

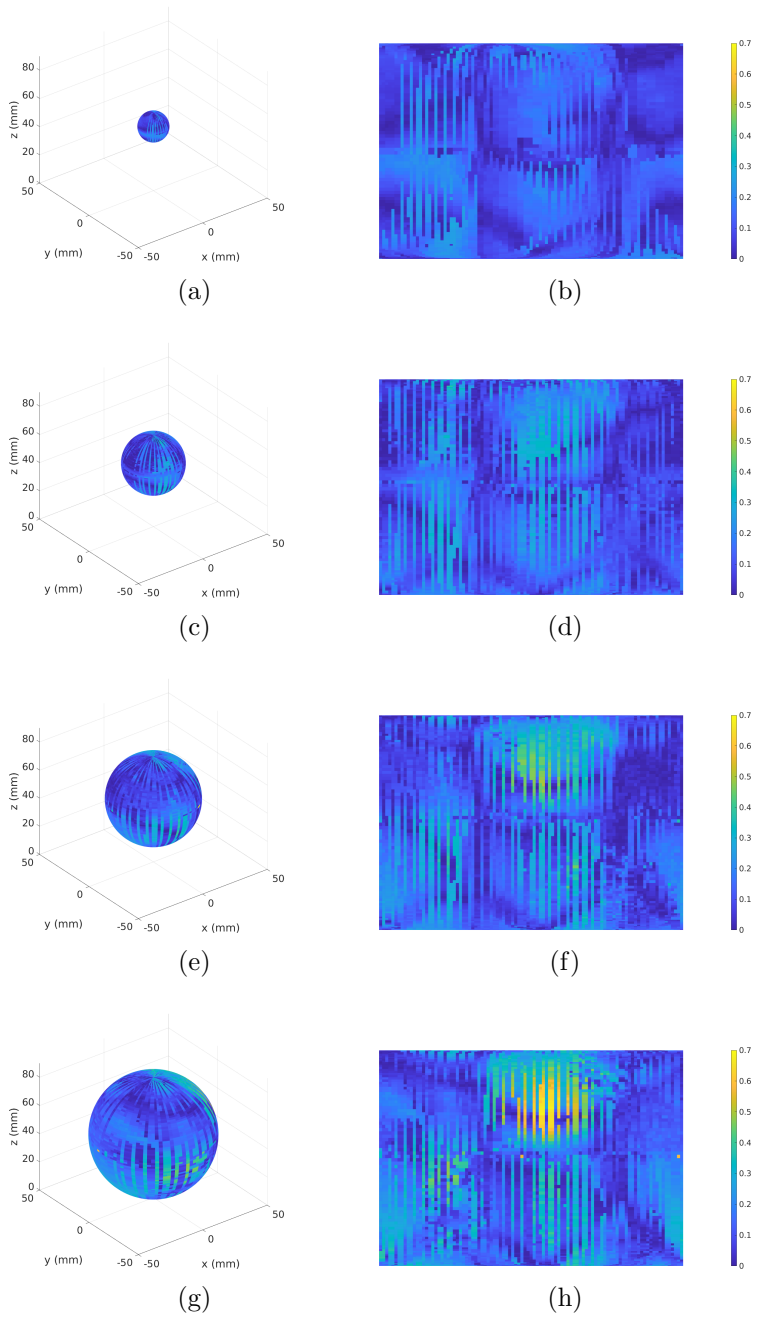


Figure 3.5: Absolute error of the dual arm manipulation through the workspace. The workspace has been projected using the Lambert equal-area cylindrical projection, the error is reported in mm. a) the workspace surface of the sphere with radius 10 mm, b) the projected surface of the sphere with radius 10 mm, c) the workspace surface of the sphere with radius 20 mm, d) the projected surface of the sphere with radius 20 mm, e) the workspace surface of the sphere with radius 30 mm, f) the projected surface of the sphere with radius 30 mm, g) the workspace surface of the sphere with radius 40 mm, h) the projected surface of the sphere with radius 40 mm.

Table 3.2: The positioning error between the PSM1 and PSM2 during the dual-arm manipulation experiment

Radius (mm)	Max error (mm)	Mean error (mm)	Std dev (mm)
10	0.61	0.11	0.06
20	0.37	0.13	0.08
30	0.51	0.14	0.10
40	0.62	0.16	0.11

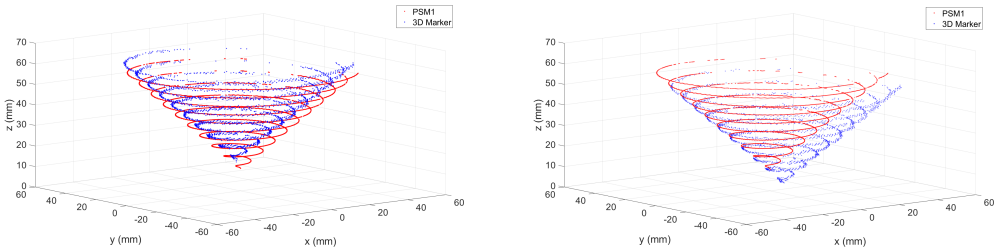
the surface of the spheres. This ensures good repeatability of motions in the whole workspace. The accuracy of our calibration method in 3D is compatible with the requirements of surgery (the mean error between the arms is below 1 mm, comparable with the known kinematic accuracy of the da Vinci®).

2D/3D projection

In the last scenario the PSM1, with a colored marker on its tip, executes a spiral-shaped trajectory along the entire workspace. The RGB-D camera identifies the marker in the image plane, and the corresponding 3D point can be computed using the depth value. The trajectory starts near the origin of the common reference frame and then increases in radius and height according to the following parametric equations

$$\begin{cases} x(t) &= \kappa t \cos(\omega t) \\ y(t) &= \kappa t \sin(\omega t) \\ z(t) &= \kappa t \end{cases}$$

where ω is the constant angular speed and $\kappa \in \mathbb{R}$ is a time-scaling factor. The orientation of the end-effector is kept fixed towards the camera along the trajectory. We measure the Euclidean error between the points in the trajectory executed by the arm and the re-projected points from the camera image plane. Figure 3.6 and Table 3.3 show that the re-projection accuracy with our method significantly outperforms the one reached with Tsai's. In fact, the mean error (4.71 mm) and the maximum error (11.76 mm) are four and two times smaller than the ones achieved by Tsai's method. It is important to remark that the measured error also includes the marker detection accuracy. Finally, Figure 3.7 shows the re-projection of PSMs end-effector position onto the camera image plane with both calibration methods. Our method achieves a better re-projection of the 3D instruments.



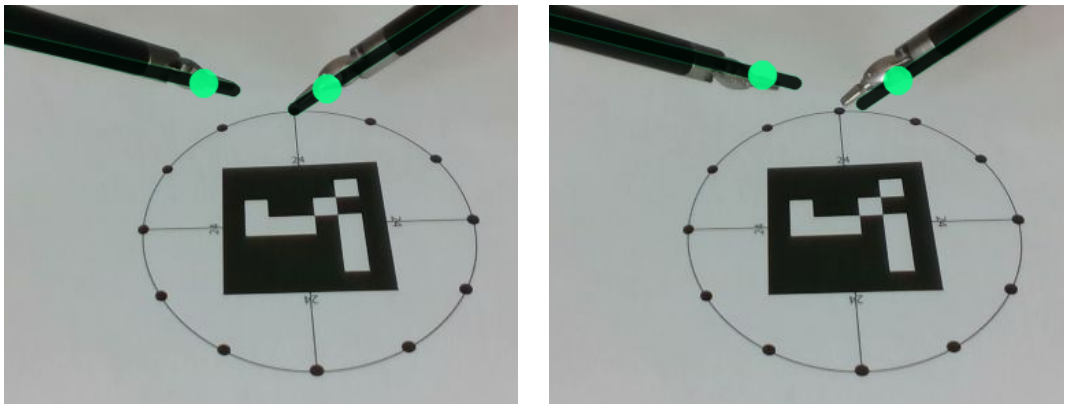
(a) Proposed method

(b) Tsai method

Figure 3.6: Spiral-shaped trajectory executed by the PSM1 with our method in (a) and Tsai's method in (b). The red trajectory represents the kinematics of the PSM1, while the blue trajectory represents the marker identified in 3D space.

Table 3.3: A comparison of the error between the marker tip trajectory and the measured tip trajectory for the projection test

	Max error (mm)	Mean error (mm)	Std dev (mm)
Our method	11.76	4.71	0.89
Tsai [57]	20.85	16.41	1.21



(a) Our method

(b) Tsai's method

Figure 3.7: An example of re-projection of da Vinci® surgical instruments by using kinematic re-projection of the model directly onto camera color image.

3.3 Rigid 3D registration of pre-operative data

In the previous section we proposed an accurate method to compute a shared frame for a robotic system using an RGB-D camera. We want now to use such calibration to provide specific target points and volumes of interest (VOI) by exploiting a novel registration methodology for pre-operative data. These VOIs represent bounding regions where the autonomous system has to look to detect target objects which can be later translated into targets in real-time. With the proposed workflow, we can register an anatomical model extracted from a pre-operative model (CT or MRI) with a sparse or dense 3D reconstruction depending on the type of vision system available.

The registration of the 3D model is based on an initial 3D feature matching and later refining using the Iterative Closest Point (ICP) algorithm. Once the registration is available the pre-computed regions can be mapped in the reference frame of the robot's arms and can be used to provide way-points during the surgery. Since the registration we are going to obtain is rigid, and since the pre-operative model is not deformed during the procedure we can use such methodology to extract only anatomical information which can be assumed to be fixed or subjected to small deformations over time.

Even if the shared reference frame is build using a 3D sensor we will assume to reconstruct the environment using a monocular camera and applying SLAM. This is because RGB-D endoscopes are not already available on the market and so we cannot rely on them for any intra-operative scan. Moreover, even if stereo endoscopes are available and currently used in all the R-MIS scenarios, we can use only one channel of the da Vinci® stereo endoscope. This is because it allows to reconstruct larger areas with respect to those that can be obtained with the da Vinci® stereo camera. In fact, the small baseline (around 5 mm) would require the endoscope to be placed too close to the surface (approximately 5 cm) in order to achieve a reliable reconstruction.

The overall architecture is depicted in Figure 3.8. The pre-operative model is obtained from an MRI scan of the anatomical phantom used in RARP experiments, see Figure 2.11b. In particular, the semi-automatic segmentation approach provided by ITK-SNAP framework [61] is exploited to extract the structures of interest. In order to register the pre-operative model (Figure 3.9) to the partial view provided by the endoscope, we perform an initial step to extract the visible surface from the complete model, which is the only portion that can be aligned with the camera view. Removal of occluded and unreachable parts is possible if an a priori estimation of the ECM pose with respect to the operational area is available. Since the endoscope motion is restricted

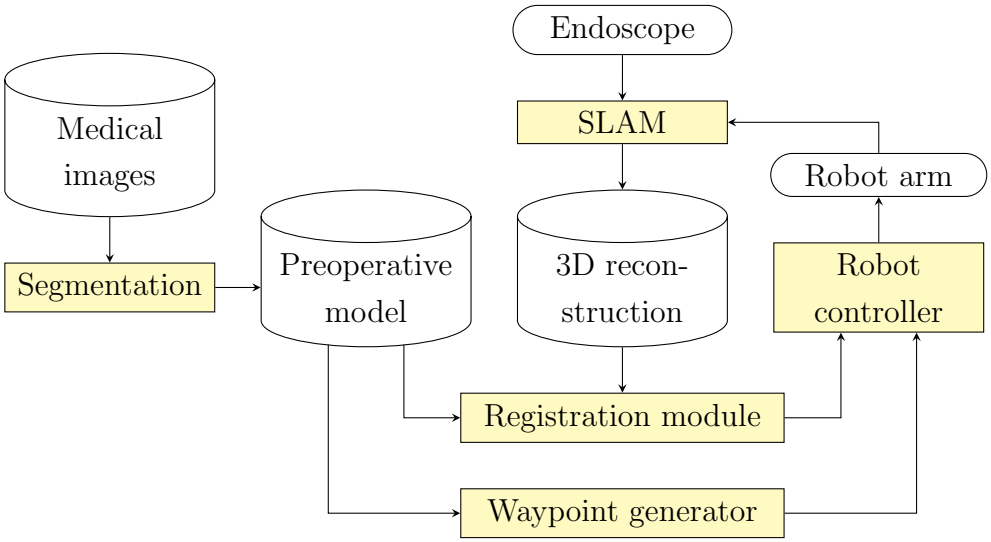


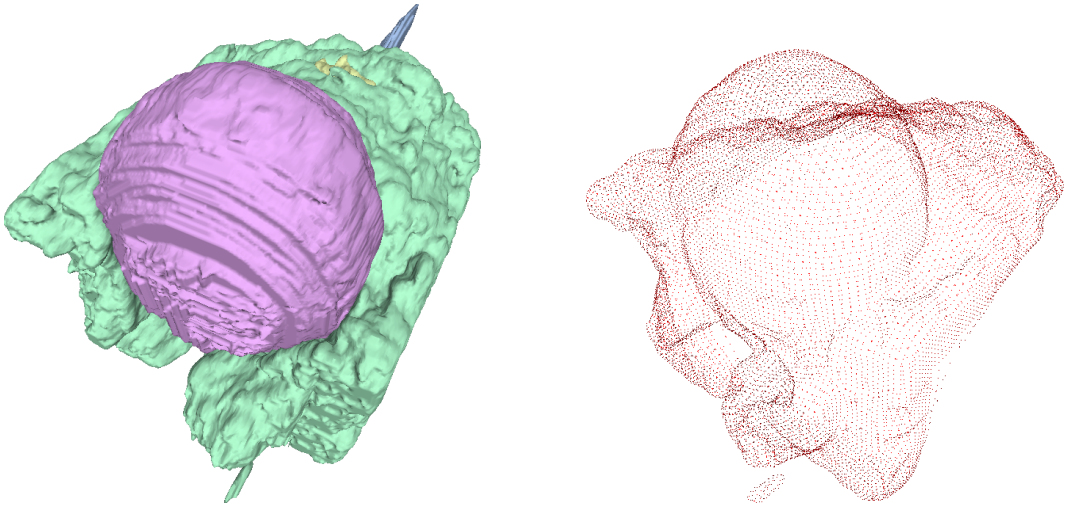
Figure 3.8: The proposed software architecture for the 3D rigid registration and waypoint generation using a monocular camera.

by the RCM of the camera holder, it is possible to manually select and discard the parts of the model which cannot be seen.

Once the portion of the MRI used for the registration is defined, as shown in Figure 3.9b, we use the Poisson surface reconstruction approach [62] to smooth the artefacts introduced during the segmentation and we recompute the normal per vertex on the aforementioned surface. The final 3D pre-operative model is converted into a point cloud representation discarding the faces from the mesh.

3.3.1 Scaled ORB-SLAM

The most important part of a SLAM system is the initialisation step. It is necessary to define a coordinate system for camera pose estimation and 3D reconstruction in an unknown environment. Therefore, in the initialisation, the global coordinate system should be first defined, and a part of the environment is initially reconstructed in the global coordinate system. For monocular SLAM it is harder, because the depth cannot be recovered from a single image frame. To initialise the map, ORB-SLAM computes two geometrical models: a homography assuming a planar scene, and a fundamental matrix assuming a non-planar scene. The map initialisation is done when the two-view configuration is reliable, detecting low parallax cases and well-known twofold planar ambiguity [63], otherwise the initialise map would be corrupted. Once the map initialisation is started the first step of the algorithm consists in finding the initial correspondences $(\boldsymbol{x}_c, \boldsymbol{x}_r)$ between the current frame F_c and the reference



(a) 3D model (b) Point cloud

Figure 3.9: The anatomical model extracted from the MRI. a) the segmented 3D model, b) the final point cloud used in the registration phase.

frame F_r extracting the ORB (Oriented FAST and Rotated BRIEF) features, which are rotation invariant and robust against noise. When there are enough matches, the initialisation procedure starts to compute a homography $\boldsymbol{H}_{cr}$ and a fundamental matrix $\boldsymbol{F}_{cr}$ using the following equations

$$\boldsymbol{x}_c = \boldsymbol{H}_{cr}\boldsymbol{x}_r, \quad \boldsymbol{x}_c^T \boldsymbol{F}_{cr}\boldsymbol{x}_r = 0 \tag{3.6}$$

with the normalised Direct Linear Transformation (DLT) and 8-point algorithms, respectively, as explained in [64]. At each iteration it computes a score $S_{\boldsymbol{M}}$ for each model $\boldsymbol{M} \in \{\boldsymbol{H}, \boldsymbol{F}\}$, $\boldsymbol{H}$ for the homography, $\boldsymbol{F}$ for the fundamental matrix. Afterwards a model is chosen according to this equation

$$R_{\boldsymbol{H}} = \frac{S_{\boldsymbol{H}}}{S_{\boldsymbol{H}} + S_{\boldsymbol{F}}} \tag{3.7}$$

where $R_{\boldsymbol{H}}$ is the robust heuristic. If $R_{\boldsymbol{H}} > 0.45$, the selected model is the homography, otherwise the fundamental matrix. If the scene is planar it can be explained by a homography, otherwise it is selected the fundamental matrix. Using the selected model, we obtain the estimation of both the pose and the motion of the camera, thus the map reconstruction can be started. To estimate the scaling factor we add to the standard initialisation the kinematic measurements of the ECM. During the initialisation phase we keep track of the real position of the camera with respect to the remote centre of motion of the ECM.

Let $\boldsymbol{P}_i$ and $\boldsymbol{P}_f$ be the homogeneous transformation of the camera pose, with respect to the ECM base frame, registered at the beginning and at the

end of the correspondence matching of ORB-SLAM. The first virtual camera pose of the SLAM is no longer the one estimated during the correspondence matching but is substituted by the relative transformation of $\boldsymbol{P}_f^i = \boldsymbol{P}_i^{-1}\boldsymbol{P}_f$. Then assuming zero rotation during the initialisation phase, the scaling factor s can be computed as the ratio of the measured translation $\boldsymbol{t}_f^i$ and the translation $\hat{\boldsymbol{t}}_f$ estimated by the SLAM

$$s = \frac{|\boldsymbol{t}_f^i|}{|\hat{\boldsymbol{t}}_f|}. \tag{3.8}$$

The translation vectors $\boldsymbol{t}_f^i$ and $\hat{\boldsymbol{t}}_f$ are extracted from the homogeneous matrix $\boldsymbol{P}_f^i$ and $\hat{\boldsymbol{P}}_f$, where $\hat{\boldsymbol{P}}_f$ is the position of the virtual camera computed by the SLAM during the initialisation phase. Finally, we apply the transformation $\boldsymbol{P}_f^i$ and the scaling factor s to the initial scene reconstruction in order to keep the same relative distance between the points and the new camera position. The assumption of zero rotation during the initialisation phase ensures that the relative motion of the 2D ORB features detected by the correspondence matching is due only to a translational movement and so (3.8) is the desired scaling factor.

3.3.2 Registration procedure

The registration is based on a feature-based initial alignment followed by a non-linear least squares minimisation of the point-to-plane distances between the two point sets (the SLAM 3D reconstruction and the 3D pre-operative model). To prevent convergence issues related to the different spatial sampling, we discretize both the SLAM map and the pre-operative model with the same step size in order to have the same spatial point density.

Initial alignment

The initial alignment of the two point clouds is required because the ICP algorithm converges easier to a feasible solution if the input point clouds have been already partially aligned. The partial alignment is done using a feature based correspondence grouping which provides an initial transformation matrix $\boldsymbol{M}_g$ applied later to the source point cloud to increase the robustness of ICP. From the two point clouds we extract two sets of Intrinsic Shape Signatures (ISS) key-points: K_s for the source point cloud and K_t for the target one [65].

The ISS key-points are local descriptors which are generally used in applications like registration, object recognition, categorisation and are known to be stable, repeatable and discriminative. An ISS is obtained counting the

weighted sum of points laying in a local 3D histogram built in a spherical angular space constructed around each feature point. For each point cloud the initial set of ISS key-points is encoded in a set of vectors using the Fast Point Feature Histogram (FPFH) which allows robust multi-dimensional descriptor of the local geometry around a point [66].

To estimate $\boldsymbol{M}_g$, a correspondence set C must be computed. Let $C = \bigcup_i C_i$ where C_i is the correspondences set for each vector $\boldsymbol{f}_i$ in the source feature set. The correspondences set is composed of the nearest neighbours of $\boldsymbol{f}_i$ in the FPFH feature space. The set C is then refined applying a cascade of correspondence rejection methods. First of all we enforce a normal direction matching and subsequently on the resulting subset we make the correspondence injective applying a duplication filtering which keeps only the closest neighbour. Finally C is further refined applying Random Sample Consensus (RANSAC) to estimate a transformation between the two correspondences set. The elimination of outlier correspondences is based on the Euclidean distance between the points once the computed transformation is applied to the source point cloud. The final transformation $\boldsymbol{M}_g$ is computed on the filtered set C using SVD.

Iterative closest point

The ICP algorithm exploiting the point-to-plane method provides a more robust and much faster convergence [67] than the classical position-based implementation. The poin-to-plane approach differs from the standard ICP technique since it minimises the distance between the source point $\boldsymbol{s}_i$ with the plane defined by the target point $\boldsymbol{t}_i$ and its normal $\boldsymbol{n}_{\boldsymbol{t}_i}$

$$\boldsymbol{M}_{opt} = \arg\min_{\boldsymbol{M}} \sum_{i=1}^{N} ((\boldsymbol{M}\boldsymbol{s}_i - \boldsymbol{d}_i)^T \boldsymbol{n}_{\boldsymbol{t}_i})^2 \tag{3.9}$$

where $\boldsymbol{M}_{opt}$ is the transformation matrices which aligns the source point cloud to the target one. The source points $\boldsymbol{s}_i$ in our method are previously transformed according to the initial alignment procedure using the transformation matrix $\boldsymbol{M}_g$ such that $\boldsymbol{s}_i = \boldsymbol{M}_g \bar{\boldsymbol{s}}_i$ where $\bar{\boldsymbol{s}}_i$ are the raw measured points. Given a source, i.e., actually the SLAM output, and a target point cloud, i.e., the pre-operative model, each iteration of the ICP algorithm establishes a set of pair-correspondences between points. The output of an ICP iteration is the 3D rigid-body homogeneous transformation $\boldsymbol{M}$ that aligns the source points to the target point cloud such that the total error between the corresponding points is minimised. As a rule of thumb in case of registration of dense to

Table 3.4: Overall system accuracy evaluation, the position of the points are expressed in $\mathbb{R}^3$.

Landmark	MRI (x_m, y_m, z_m) (mm)	Reference (x_r, y_r, z_r) (mm)	Error ϵ (mm)
1	6.33, -60.41, 62.99	9.00, -58.00, 59.00	5.4
2	7.59, 22.05, 48.37	6.00, 29.00, 52.00	8.0

sparse point cloud, ICP performs better if the source point cloud is the sparse one. In the case of 3D pre-operative model the normal vectors are precomputed, while they are computed before the registration routine for the SLAM point cloud. The estimation process is also taking into account the camera position in order to have all the normal vectors pointing towards it.

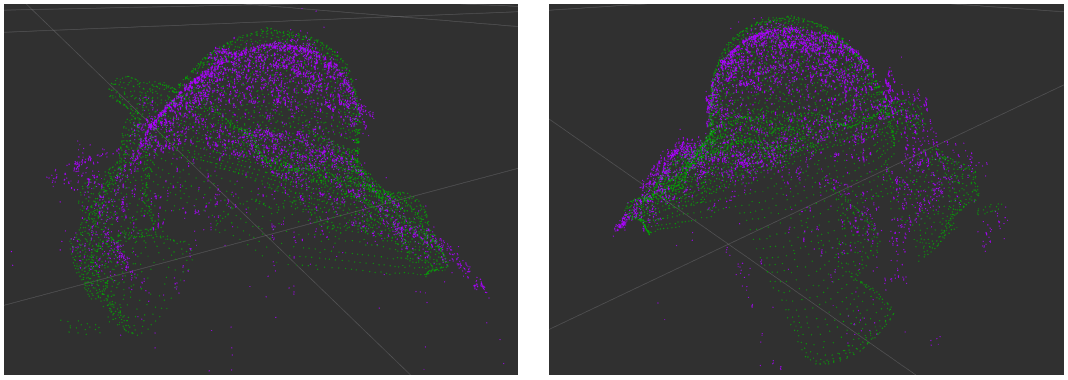
3.3.3 Scaling evaluation and error bounds

To evaluate the accuracy of the automatic scale estimation we compared the size of the visible anatomy of the RARP phantom both in the 3D reconstruction and in the MRI (assumed to be the ground truth).The resulting error ϵ_s due to the estimation of the scaling factor s is of 4 mm and is due to the combination of the 3D reconstruction error and the hand-eye calibration $\boldsymbol{T}_e^c$.

We then evaluated the accuracy of the 3D registration technique by comparing the position of a set of fiducial points in the MRI with the position of the robot end-effector while touching them on the phantom. The accuracy is evaluated in terms of the position error $\|\epsilon_i\|_2$

$$\epsilon_i = \begin{bmatrix} x_m^i & y_m^i & z_m^i \end{bmatrix}^T - \begin{bmatrix} x_r^i & y_r^i & z_r^i \end{bmatrix}^T$$

where (x_m^i, y_m^i, z_m^i) is the i-th fiducial point selected on the surface of the 3D model, and (x_r^i, y_r^i, z_r^i) is the corresponding i-th position of the end-effector. Since our phantom doesn't have any specific landmarks in it, we selected the corners along the boundary of the fat as fiducial points. The arm used to perform this measurement is the patient side manipulator (PSM) of the da Vinci® which has a position error of 1 mm with respect the common reference frame and is assumed to be the ground truth. The average error ϵ is around 6.7 mm. It is worth remarking that the errors ϵ_i reported in Table 3.4 include the scaling estimation error ϵ_s and the uncertainties due to the multi-arm 3D calibration methodology proposed in the previous section. In fact, the 3D reconstruction obtained by the SLAM is referenced to the camera reference frame and in order to be placed in the common reference frame (world) we need to apply multiple



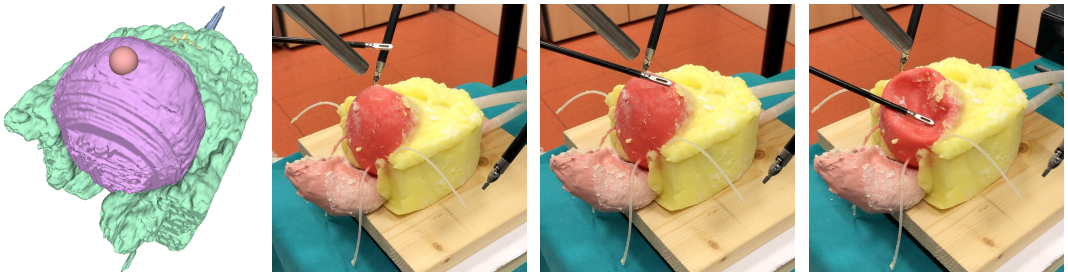
(a) (b)

Figure 3.10: The resulting registration of the pre-operative map with the anatomy 3D reconstruction. The purple dots are part of the point cloud obtained by SLAM, the initial map is shown in green.

transformations. The 3D registration accuracy is so the combination of: (i) the shared reference frame calibration accuracy $\boldsymbol{T}_w^{eb}$, $\boldsymbol{T}_w^{pb}$ (which affects the ECM and the PSM1 end-effector position), (ii) the hand-eye calibration accuracy $\boldsymbol{T}_e^c$ (which affects the positioning of the endoscope with respect to the ECM end-effector), and (iii) the ICP registration accuracy. The Euclidean norm of the misalignment between the source point cloud (pre-operative model) and the target point cloud (SLAM model) is of 0.560 mm.

3.4 Qualitative and functional evaluation

To evaluate the functional effectiveness of the proposed methodology we performed autonomously the task *pushing down the bladder* from the RARP procedure. This task is needed to create the free space for the main surgeon to resect the prostate. The control architecture of the autonomous arm receives the target points defined on the pre-operative 3D model. The task of pushing down the bladder can be modelled, in the first place, by a simple finite state machine (FSM). The FSM is composed of four states: approach the apex of the bladder (S0), push down the bladder (S1), stay still (S2), and leave motion (S3). The "approach point" is defined as the highest point of the bladder with respect to the MRI reference frame. The target position, reached during the *push down* motion, is obtained applying a vertical displacement starting from the approach point. Since in the surgical environment there are also the da Vinci® instruments, the trajectory of the assistant robot must be collision-free. Figure 3.11 shows a few snapshots taken during the *bladder pushing* experiment. The "approach point" selected by the system is depicted



(a) Selected point (b) Initial pose (c) Bladder approach (d) Bladder push

Figure 3.11: An example of the bladder pushing phase. a) the red sphere represents the point selected directly on the MRI, in this case the apex of the bladder, b) the robot initial position, c) the robot end-effector once it has reached the approach position over the bladder, d) the robot end-effector at the target position.

in Figure 3.11a: after the registration phase, the position is transformed in the robot reference frame and sent to the robot. As shown in Figure 3.11c the end-effector of the robot is placed nearby the desired point. Once the point on the surface of the bladder is reached, the next step of the procedure is to move towards a point placed inside the bladder in order to push the organ down. The system calculates that point taking a vertical displacement of 0.045 m along the vertical axis starting from the approach point. Figure 3.11d shows the robot end-effector at the desired target position.

This preliminary experiment validates the proposed multi-arm calibration algorithm and the pre-operative registration. This allows us to proceed with the definition of a supervisory controller for the complete RARP procedure which will be presented in Chapter 4.

Chapter 4

Supervisory control

In the previous chapter we introduced the computer vision technologies needed to let a robotic system understand and interact with the anatomical environment. In this chapter we focus on the exploitation of such technologies to define a supervisory controller methodology to model and plan the execution of a surgical procedure. The solution proposed is based on a *revised statechart model*. The statechart formulation has been selected for its inherent combination of structured and unstructured knowledge that can be handled hierarchically at different levels of abstraction. To achieve this abstraction, the statechart at the top level is split in two concurrent regions: procedure and observer. The *procedure region* represents the medical knowledge extracted from clinical trials with surgeons and from literature review; the *observer region* is composed of a concurrent set of FSMs that provides a logical description of the environment state (e.g., kinematics state, arm-anatomy contact state, etc.). Such concurrent observers would be the way we used to integrate machine learning techniques into the supervisory controller. We will consider them as a set of soft-sensors where each sensor will be a separate observer which triggers events. The decision on how such triggers drive the procedure evolution over time is controlled by the structure of the procedure region. The procedure region of the statechart is subdivided in hierarchical levels to refine how the desired behaviour of the robotic system is defined. Such division of knowledge at the procedure design level helps in simplifying both modelling and comprehension of the statechart itself. The control commands for the robots are then defined in the innermost level.

Part of this chapter is based on the following publication:

› F. Falezza, N. Piccinelli, G. De Rossi, A. Roberti, G. Kronreif, F. Setti, P. Fiorini, and R. Muradore, "Modeling of surgical procedures using statecharts for semi-autonomous robotic surgery," *IEEE Transactions on Medical Robotics and Bionics*, vol. 3, no. 4, pp. 888–899, 2021

In case of unexpected events, like unmodelled behaviours or mechatronics malfunctioning, the proposed supervisory controller will delegate the safety of the patient to the surgeon. This solution will let a human operator to always intervene using bilateral teleoperation. In particular, in Chapter 5 and Chapter 6 a revised version of hierarchical passivity-based teleoperation and a novel solution to such problem will be presented.

4.1 State of the art

Two different approaches can be adopted to model the medical knowledge: a top-down and a bottom-up approach. The *top-down approach* is based on encoding prior knowledge into a formal representation understandable by both humans and machines. Different approaches have been proposed, like description logic [69], formal ontologies [70], or defeasible reasoning [71]. Statecharts models are a graphical specification formalism that allows the nesting of Finite State Machines (FSMs hierarchy), their orthogonality (FSMs parallelism) and re-usability of components [72], [73]. The major advantage brought by FSMs is that they can be formally verified [74], and, therefore, are always guaranteed to operate according to their design. For this reason, FSMs are widely employed in the representation of mission-critical workflows, such as the case for surgical procedures. The representation power of statecharts has been exploited to build a discrete-event simulation model of the pre-operative process [75]. The recovery progress of each patient through surgical care is described as a series of asynchronous updates in patients' records; these updates are triggered by events produced by parallel FSMs that represent concurrent clinical and managerial activities. The *bottom-up approach* infers a model from raw data through data analysis techniques, such as deep learning, possibly in an unsupervised, end-to-end manner to speed-up the process and to avoid labelling bias [76].

In this work, we adopt a safer approach that follows the *engineering stack* guidelines for which the top-down model is adapted in its formulation to the events based on the available observations from both the environment and the robots [77]. This improves safety and explainability of, e.g., Machine Learning (ML) and Deep Learning (DL) modules used to process the data and extract relevant information from video streams [78], and sensor measurements (e.g. kinematics in case of robotic surgery) [79]. The work in [80] adopts multiple disjointed FSMs, one for each surgical subtask, where the parameters (i.e., the thresholds viable to trigger events) are learned in a reinforcement learning

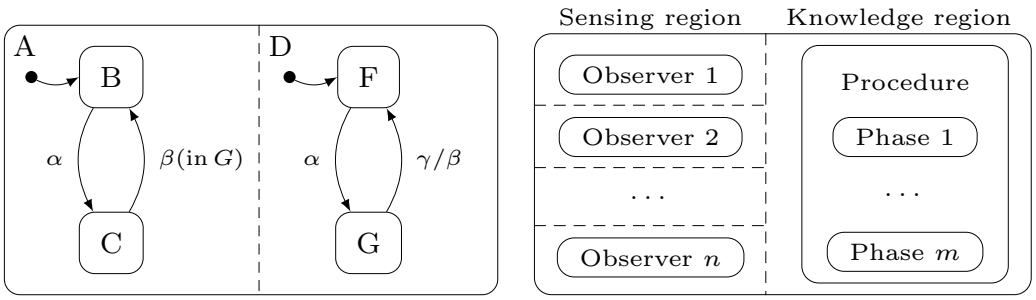
manner.

To automate basic surgical tasks, many approaches have been proposed. For instance, the description using Finite State Machines (FSM) can be used to account for simple tasks, where the environment is assumed to be static and situation awareness can be neglected. In [81] an automated mechanical needle guide to improve precision is proposed. In [82] an FSM based framework for the automation of surgical sub-tasks is developed and tested on simple surgeon training tasks like peg-and-ring and knot-tying. In [83] a depth-sensor has been used to increase the accuracy of peg-and-ring task. Finally, in [84], a Hierarchical Finite State Machine (HFSM) is used to control autonomous mobile systems. The last work shows how the hierarchy can be exploited to subdivide the controller between its discrete and continuous-time parts, allowing the separation of the high-level decision-making to their low-level implementation.

An alternative approach to FSM is represented by Behaviour Trees (BT) with their emphasis on modularity [85]. In [86] a BT is used to model and control a semi-autonomous simulated brain tumor ablation; here the leaves of the BT represent the surgical sub-tasks of the ablation. The main drawback of BTs is the lack of introspection. Indeed, the current state of the system cannot be retrieved directly, but it must be derived analysing the path from the root node to the current running leaf. In [50] a cognitive framework to perform autonomous needle insertion has been proposed. The supervisory controller was built by using a Hidden Markov Model (HMM).

Unfortunately, all of these data-driven approaches, require a large amount of data to achieve a robust learning, which is not usually available in surgery. Moreover, the planned action cannot be easily interpreted and monitored by a human expert since data-driven models are often based on latent variables representation of the environment.

In critical scenarios like surgical procedures, knowledge-based approaches are preferred since they provide a clearer description of the workflow. For instance, in [87] an ontology-based framework for the automation of the peg&ring task has been proposed. The main drawback was the lack of real-time reconfiguration of the system. In fact, ontologies are much more used in the field of situation understanding by humans [88]. A solution to overcome the limitation of ontologies can be the non-monotonic programming, where the planning is carried out in a more flexible way, thus the knowledge can be updated in real-time from the sensing information. In [7] Answer Set Programming (ASP) has been used to define the reasoning module and has been successfully applied on an automated peg-and-ring task. The drawback of non-monotonic program-



(a) Example of statechart (b) Revised statechart structure

Figure 4.1: The graphical representation of a statechart. a) an example of a two-level and concurrent statechart, b) the revised statechart is composed of n observers that run concurrently to the procedure. The observers generate events consumed by the procedure's state transitions. The procedure hierarchy is not reported in this figure but is composed of three levels.

ming resides in the computational complexity required to solve a planning problem, which makes this approach often unsuitable for real-time applications.

4.2 Modelling methodology

As mentioned above, the proposed modelling methodology relies on the statecharts visual notation, a simplified example of which is shown in Figure 4.1a, that will be briefly discussed to ease the reading in the following sections. A statechart uses rounded rectangles to denote the states (or regions) at any level, using encapsulation to express the hierarchy relation. Arrows are allowed to start and terminate at any level. Each arrow is labelled with an event, optionally with a condition between brackets and an output event after a backslash, and marks the possible transitions between states. The initial state is marked with an arrow pointing to it starting from a small dot without label. The encapsulations imply the exclusive-or (XOR) decomposition of inner states, meaning that the current state must be only one and it must be selected from the states set of the grouping state. Moreover, the encapsulations could introduce the AND decomposition, capturing the property that, being in a state, the system must be in all of its AND components using dashed lines. The statechart notation is too much expressive for our case study as it allows modelling behaviours that are not required or admissible in our application. Therefore, in this thesis we propose a revision of statecharts which imposes a clear separation between the sensing of the environment and the representation

of the procedure knowledge into two distinct *regions* of the chart, as shown in Figure 4.1b. Indeed, the *concurrency* capability of the statecharts will be exploited only to represent the parallelism of the sensing, while the *hierarchy* will be used only to separate the medical knowledge from the implementation of the robotic tasks. The proposed revisions is based on the following additional rules:

R_1 The hierarchy depth of the procedure must be equal to three: *activity*, *task* and *primitive* (see Section 4.2.1)

R_2 At the lowest level (i.e., primitive), the states are atomic and hierarchy-agnostic, thus they do not depend on the execution time or the order with respect to others states (see Section 4.2.2)

R_3 The sensing region is evaluated concurrently to the knowledge region, it is the only generator of events during transitions (see Section 4.2.3)

R_4 Concurrency is only allowed within the sensing region, thus transitions between concurrent FSMs are forbidden (see Section 4.2.4).

Using this revised statechart modelling, we can now define the adopted heuristics for the planning. We exploit the intrinsic priority of the hierarchy since an event generated by the sensing system can be adopted as a transition trigger by any level of the procedure. At each control cycle, the supervisory controller, which evaluates the statechart, starts from higher to lower level and checks if there exists a trigger event on the edges exiting the current state. If a trigger event is present, then the state is activated and the search is stopped. When a transition occurs, the innermost FSMs are reset to their initial state. Even if such heuristic ensures the high-level command to be prioritised, it can lead to transitions shadowing, meaning that transitions on lower-levels could never been crossed if transitions at higher-levels on the same trigger are fired. To minimise such effect, the modelling of the procedure should be approached exploiting a semantic separation of events. At the higher levels of the procedure, the events should refer only to the environment conditions for the activity (e.g., in surgery, positioning over the bladder, thread cut, catheter visible, etc.), while at the lowest level the events should refer only to the robot internal state (e.g., target pose reached, pose not reachable, grasping, etc.). The extension to multiple robots can be obtained by simply duplicating the statechart and adjusting it for each manipulator. The synchronous operation of multiple statecharts that operate on the same triggers is guaranteed by the assumption that each trigger produced by every observer is processed simultaneously by all statecharts. Therefore, each robot effectively observes the

environment, including all others robots, independently, thus eliminating the necessity of specific synchronisation states.

4.2.1 Procedure depth

Generally, the number of hierarchical levels allowed in a statechart is unbounded. In our case, as stated in the rule R_1, the number of the levels is set to three. The adoption of a three-level hierarchy has been proposed in [89] and is also part of the definition of the Hierarchical Task Network (HTN) [90], which is a popular task planning methodology. Therefore, the resulting procedure can be defined by grouping the states of the statechart into three well-defined levels of a surgical procedure: **phase**, **action** and **surgeme**, mapped respectively into *activity*, *task*, and *primitive*, which is a lexical formalism also adopted in [91].

The highest level, composed of phase states, is the equivalent to the goals of the STRIPS modelling [92]. In the surgical field, this level models the main phases of a complex surgical procedure (e.g., in the case of radical prostatectomy, some goal tasks could be bladder mobilisation, bladder neck transection, or an idle statement). The middle level, composed of action states, can be seen as a set of intermediate tasks that compose the goal task (e.g., for the bladder neck transection phase some actions are grasping the catheter and pulling the catheter). Finally, in the lowest level we find atomic sub-tasks that cannot be logically subdivided into smaller subsets, thus composed of surgeme states (e.g., close the gripper or move to a specific point).

4.2.2 Atomic and hierarchy-agnostic execution

The rule R_2 enforces the re-usability of the surgemes allowing the definition of multiple complex surgical procedures by means of a relatively small set of common and shared primitives. The side effect of this design choice is the need of prohibiting any transition from a lower level to a higher one. In fact, the re-usability requires to have hierarchy-agnostic surgemes and of course transitions towards the upper levels require a knowledge about the parent states (not available at the surgeme level).

4.2.3 Event generation

An observer is an FSM that operates concurrently to the procedure with the aim of observing the environment and of generating trigger events based on measurements (observations). The definition of the observer entities allows

separating the generation of triggers, which happens in the sensing region, from their consumption, which happens in the knowledge region. Therefore, the rule R_3 states that the knowledge region must not generate triggers to avoid infinite loops, undefined behaviours, and deadlocks. The definition of the events can be obtained in two different ways:

- bottom-up: the procedure is modelled and refined by surgeons using only the available observers;
- top-down: the procedure is first defined based on the knowledge of surgeons, then the engineers will develop the requested observers to accomplish the procedure.

The formalism presented hitherto intends to logically separate these two approaches to avoid contrasts in requirements formulated by surgeons and engineers: the surgeons designing the procedure in a top-down manner could specify undetectable events (or that could require hardware that is not applicable to laparoscopy), whereas the engineers working in a bottom-up manner could overlook important events that do not rise directly from the data. Our approach simplifies the modelling of the procedure and allows to design a more efficient system based on the available sensors, surgical instruments, and computational power. The managing of the interleaved requirements should be assigned to a specialised bio-engineer who has to handle surgical and engineering critical aspects.

4.2.4 Concurrent isolation

The standard set of rules for statecharts do not prevent the definition of edges over concurrent states, but in the case of the proposed procedure-observers statecharts the transitions are meaningless as surgical procedures follow a well-defined sequence of states (which is hampered only by the occurrence of unexpected complications that must be handled case-by-case). This is the reason why we introduced rule R_4 that forbids the definition of edges on concurrent states, thus between observers and procedures, and between different observers.

4.3 Case Study: Radical Prostatectomy

In the following paragraphs, we will describe the statecharts used to control both manipulators of the *SARAS* SOLO-SURGERY platform. These charts

represent a single instance of operation modelled on the experimental setup: they demonstrate the general methodology for merging the top-down medical knowledge with the bottom-up data-driven knowledge. Initially, we will focus on the sensing part describing in detail which observers have been integrated and which events have been observed. Later, the surgemes will be defined and integrated into the platform. Finally, the procedures for the left and right arms will be modelled following the proposed methodology. The procedures schemes are shown in Figures 4.2 and 4.3, with each symbol explained in Tables 4.1 and 4.2.

We have distributed the activities of the assistant between the two robotic arms: the left side arm is in charge of the bladder neck transection phase, provided with a grasper tool for mobilizing the catheter, while the right side arm takes care of the bladder mobilization and anastomosis with a pair of curved scissors, which are curved upward during the former phase to avoid puncturing the bladder. All motions are intended to be executed with collision-free trajectories considering both the main surgeon's instruments and the anatomy as obstacles. The anastomosis for the assistant consists only in cutting the thread, which translates into motions toward the thread and then closing the jaws to cut.

4.3.1 Observers

Each *robot observer* is a statechart composed of 3 concurrent FSMs. The purpose of each observer is to identify the current state of the robot and to trigger events when a desired state is reached. The observed states are the Cartesian position of the manipulator's end-effector, the rotation of the tool and the closure/opening state of the gripper. For instance, when the target rotation of the left arm is reached, the trigger t_8^l is generated; when the tool is opened or closed, the triggers t_{12}^l or t_{10}^l are generated, respectively; t_{14}^l or t_{15}^l are generated when the target pose is reachable or not. The triggers for the right arm work in the same manner, please see Table 4.2 for the full list.

The *force observer* is made of an FSM with only two states. The purpose of this observer is to measure the interaction force, by means of the strain-gauges applied on the instrument shaft, of the autonomous robotic arm to verify whether a successful grasping happened. A threshold W_t is set: if the measured force $W_m > W_t$, then the t_2^l trigger is generated while transiting from state O_1^l to O_2^l occurs, otherwise it loops on the initial state O_1^l generating the trigger t_1^l continuously.

The *catheter observer* and the *feature observer* are two concurrent FSMs

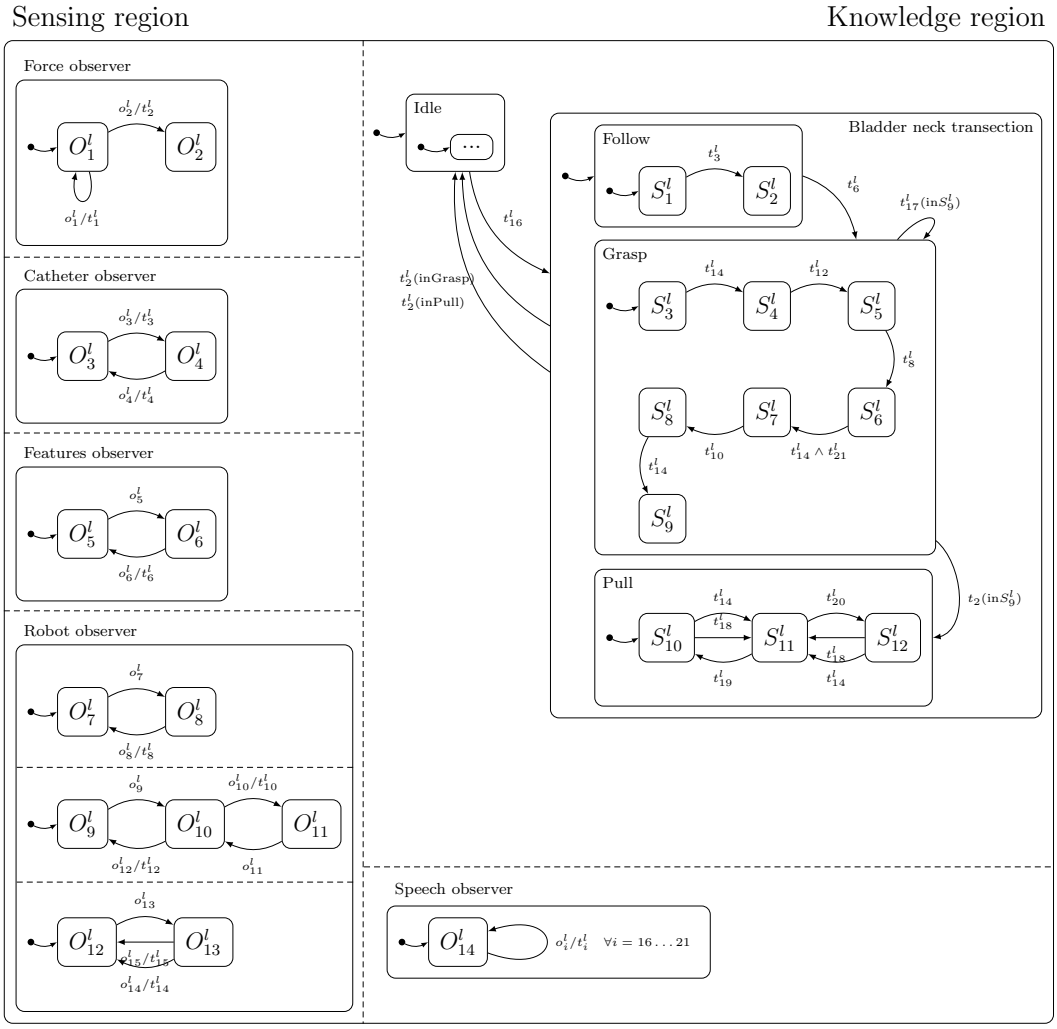


Figure 4.2: Left arm procedure modeled as statechart (i.e., bladder neck transection). The observer states are represented by O_i, the surgeme states are represented by S_j and the observed events and the triggers are labeled with o_k and t_k, respectively.

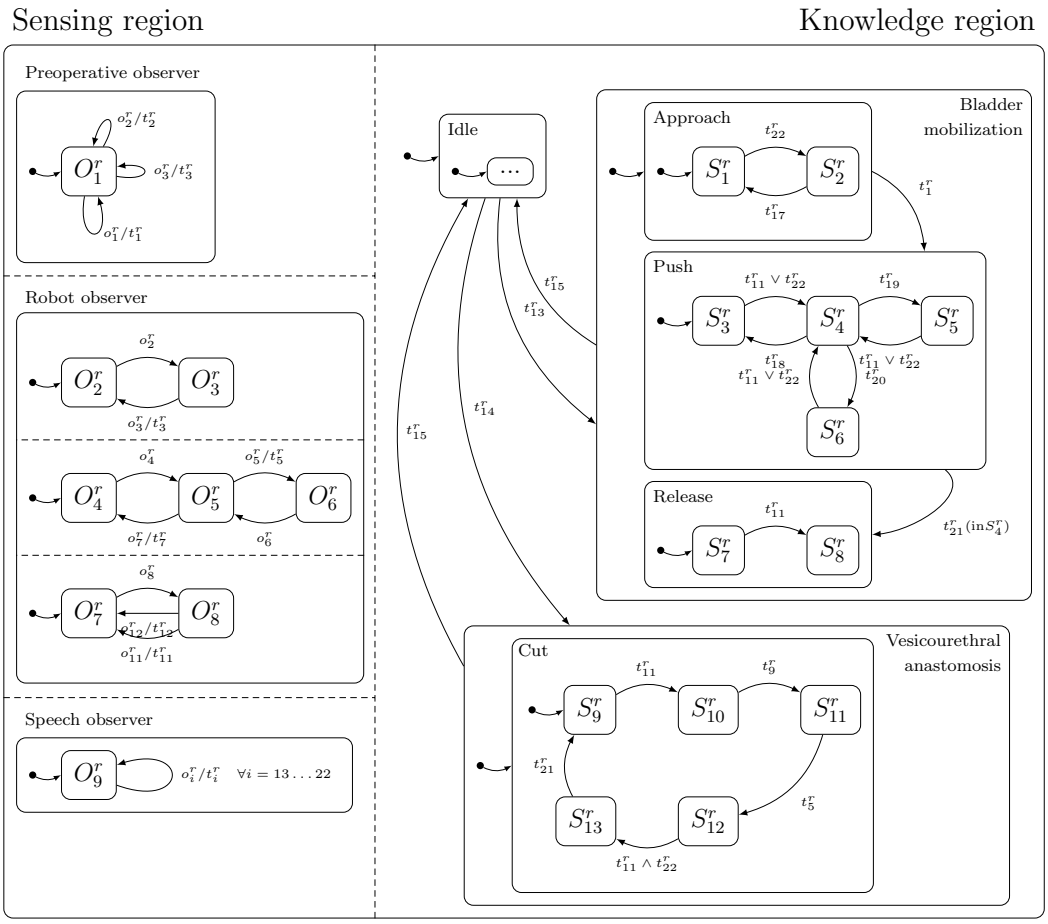


Figure 4.3: Right arm procedure modeled as statechart (i.e., bladder mobilization, vesicourethral anastomosis). The observer states are represented by O_i, the surgeme states are represented by S_j and the observed events and the triggers are labeled with o_k and t_k respectively.

Table 4.1: Description of symbols for observers, surgemes and triggers of the left arm in Figure 4.2.

Observer	Observer Label	Surgeme	Surgeme Label	Event/Trigger	Event/Trigger Label
O_1^l	Force under threshold	S_1^l	Idle	o_1^l/t_1^l	Force under threshold
O_2^l	Force over threshold	S_2^l	Follow the catheter	o_2^l/t_2^l	Force over threshold
O_3^l	Looking for catheter	S_3^l	Approach the catheter	o_3^l/t_3^l	Catheter found
O_4^l	Tracking the catheter	S_4^l	Open the tool	o_4^l/t_4^l	Catheter vision lost
O_5^l	Features under threshold	S_5^l	Rotate the tool	o_5^l	Velocity over threshold
O_6^l	Features over threshold	S_6^l	Move to the grasping point	o_6^l/t_6^l	Catheter stopped
O_7^l	Rotation in idle	S_7^l	Close the tool	o_7^l	Tool is rotating
O_8^l	Rotation moving	S_8^l	Check the grasping	o_8^l/t_8^l	Goal rotation reached
O_9^l	Tool close	S_9^l	Wait for confirm grasping	o_9^l	Tool is closing
O_{10}^l	Tool moving	S_{10}^l	Pull more	o_{10}^l/t_{10}^l	Tool closed
O_{11}^l	Tool open	S_{11}^l	Idle	o_{11}^l	Tool is opening
O_{12}^l	Robot idle	S_{12}^l	Pull less	o_{12}^l/t_{12}^l	Tool open
O_{13}^l	Robot moving			o_{13}^l	Robot is moving
O_{14}^l	Speech recognition			o_{14}^l/t_{14}^l	Goal reached
				o_{15}^l/t_{15}^l	Goal not reachable
				o_{16}^l/t_{16}^l	Speech grasp catheter
				o_{17}^l/t_{17}^l	Speech retry
				o_{18}^l/t_{18}^l	Speech stop
				o_{19}^l/t_{19}^l	Speech pull more
				o_{20}^l/t_{20}^l	Speech pull less
				o_{21}^l/t_{21}^l	Speech close instrument

Table 4.2: Description of symbols for observers, surgemes and triggers of the right arm in Figure 4.3.

Observer	Observer Label	Surgeme	Surgeme Label	Event/Trigger	Event/Trigger Label
O_1^r	Preoperative poses	S_1^r	Approach bladder	o_1^r/t_1^r	On bladder top
O_2^r	Rotation in idle	S_2^r	Idle	o_2^r/t_2^r	On bladder bottom
O_3^r	Rotation moving	S_3^r	Push bladder	o_3^r/t_3^r	On safe pose
O_4^r	Tool close	S_4^r	Hold bladder	o_4^r	Tool is rotating
O_5^r	Tool moving	S_5^r	Push less	o_5^r/t_5^r	Goal rotation reached
O_6^r	Tool open	S_6^r	Push bladder here	o_6^r	Tool is closing
O_7^r	Robot idle	S_7^r	Release bladder	o_7^r/t_7^r	Tool closed
O_8^r	Robot moving	S_8^r	Safe pose	o_8^r	Tool is opening
O_9^r	Speech recognition	S_9^r	Follow thread	o_9^r/t_9^r	Tool open
		S_{10}^r	Open scissor	o_{10}^r	Robot is moving
		S_{11}^r	Rotate scissor	o_{11}^r/t_{11}^r	Goal reached
		S_{12}^r	Approach thread	o_{12}^r/t_{12}^r	Goal not reached
		S_{13}^r	Cut thread	o_{13}^r/t_{13}^r	Speech bladder push down
				o_{14}^r/t_{14}^r	Speech cut needle
				o_{15}^r/t_{15}^r	Speech idle
				o_{16}^r/t_{16}^r	Speech close
				o_{17}^r/t_{17}^r	Speech restart
				o_{18}^r/t_{18}^r	Speech push more
				o_{19}^r/t_{19}^r	Speech push less
				o_{20}^r/t_{20}^r	Speech push here
				o_{21}^r/t_{21}^r	Speech release
				o_{22}^r/t_{22}^r	Speech stop

composed of two states with transitions that allow to loop in both states. To detect the catheter we used a template matching algorithm which finds areas of an image that match (i.e. similar) to a reference image T (called patch), which in our case is the tip of the catheter. We also restrict the search area in the endoscope image I projecting the position of the teleoperated da Vinci® arm to the camera image plane and we create a fixed size Region Of Interest (ROI) $\hat{I}$. We compare the template image against the source image by sliding it, moving the patch one pixel at a time. At each location, a metric is calculated to check how similar the patch is to that particular area of the source image. For each location of T over $\hat{I}$, the metric is stored within a matrix R. Each location (x, y) in R contains the match metric. In our experiments we used the following metric

$$R(x,y) = \frac{\sum_{x',y'}(T(x',y') \cdot \hat{I}(x+x', y+y'))}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} \hat{I}(x+x', y+y')^2}}.$$

Once a match higher than a pre-defined threshold is detected, the catheter is assumed to be found and we estimate a grasping point on it. Afterwards, salient features can be identified and tracked from frame to frame while the da Vinci® instrument is pulling up the catheter. Features extraction is performed by using Shi-Tomasi algorithm [93], which evaluates the difference in intensity for a displacement of (u, v) in all directions. For corner detection we have to maximize the following function

$$E(u,v) = \sum_{x,y} \underbrace{w(x,y)}_{\text{window function}} \; [\underbrace{\hat{I}(x+u, y+v)}_{\text{shifted intensity}} - \underbrace{\hat{I}(x,y)}_{\text{intensity}}]^2.$$

The extracted features are given to the optical flow function frame by frame to ensure that the same points are being tracked. Optical flow assumes that the pixel intensities of an object do not change between consecutive frames and the neighbouring pixels have similar motion. We used Lucas-Kanade implementation [94], which takes a 3×3 patch around each feature point to estimate its motion. The tracked features are projected in 3D by using the depth map of the RGB-D camera and then their centroid are calculated to obtain the position of the catheter in the world reference system. We also use a Kalman filter [95] to increase the accuracy of the tracking process and to handle occlusions given by tools and/or by the lack of depth information from the RGB-D camera. The model used in the filter is a position-velocity double integrator with an additional damping term to converge the velocity to zero in case of extensive occlusions. The prediction is corrected every time a measurement of the catheter's position is available.

When the catheter is found the transition from O_3^l to O_4^l is performed, triggering the event t_3^l; when the track is lost the opposite state transition is computed triggering the event t_4^l. The feature observer estimates the catheter odometry and checks whether the linear speed of the catheter goes over and then below two thresholds W_t^{max} and W_t^{min} within a sliding window T_t. This observer is used during the procedure to detect whenever the main surgeon has completely extracted the catheter from the prostate and is ready to give it to the autonomous robotic arm. Practically, if the measured speed $W_m > W_t^{max}$ and then within T_t seconds $W_m < W_t^{min}$, the transitions from O_5^l to O_6^l and then back to O_5^l are performed, triggering the event t_6^l.

An instance of the *speech observer* is available to each arm and they are FSMs composed of a single state (e.g., O_{14}^l, O_9^r). The state runs a speech recognition algorithm that adopts Natural Language Processing (NLP) to map commands by the surgeon into triggers.

The *preoperative observer* is in charge to generate a trigger when the robot's end-effector reaches a predefined position in the anatomy. The FSM is composed of a single state O_1^r, with as many transitions as the number of predefined positions. For instance, for the right arm there are three transitions t_1^r, t_2^r describing landmarks on the bladder and t_3^r which describes the idle position for the arm.

4.3.2 Surgemes

Only 4 types of surgemes have been implemented to accomplish the whole procedure under study

- *No operation surgeme* is used to keep the robot in the current state. No operation is performed until a transition is done. This surgeme is used in the states S_1^l, S_9^l, S_{11}^l, S_2^r, and S_4^r.

- *Motion surgeme* interpolates a trajectory from the current position to the target position. The target position is given as a parameter to the surgeme. If the target position is changed at run-time the trajectory is re-computed. This surgeme is used in the states S_2^l, S_3^l, S_6^l, S_8^l, S_{10}^l, S_{12}^l, S_1^r, S_3^r, S_5^r, S_6^r, S_7^r, S_8^r, S_9^r, and S_{12}^r.

- *Rotate tool surgeme* is used to rotate the tool around its main axis. It takes as argument the desired absolute rotation. This surgeme is used in the states S_4^l and S_{11}^r.

- *Open/close surgeme* is used to open or close the grasper/scissors. It takes as argument the desired target closing percentage e.g., 100% means totally closed. This surgeme is used in the states S_5^l, S_7^l, S_{10}^r, and S_{13}^r.

4.3.3 Left arm procedure

The procedure for the left arm is composed of two phases: the *idle phase* and the *bladder neck transection phase*. Since the experiment is performed in a semi-autonomous scenario, while the main surgeon is dissecting the bladder neck, the robotic assistant does not have to accomplish any specific task. For this reason, in the *idle phase* the tool stays in a safe position allowing the main surgeon to operate. In this phase, the autonomous system is waiting for the trigger t_{16}^l which corresponds to the speech command "left arm grasp the catheter", which allows the phase transition to the *bladder neck transection phase*.

The *bladder neck transection phase* is composed of three actions: *follow action*, *grasp action* and *pull action*. The first action has the purpose to follow the catheter by keeping a safe distance, while the second one is designed to grasp the catheter. The last action pulls the catheter towards the pelvis bones and handles any possible positioning adjustments requested by the main surgeon. The transition from *follow action* to *grasp action* is enabled when the catheter stops after reaching a threshold velocity (trigger t_6^l). The *grasp action* progressed only into three possible transitions which depend on the internal current state S_9^l: if t_{17}^l is triggered by the surgeon command "left arm retry", the effect is a reset of the internal state to the initial one of the action; if the measured force on the tool is under threshold, t_1^l is triggered and the state doesn't change as it waits for a slight pull by the surgeon to verify the grasp; finally, if the force is above the threshold (t_s^l), the current action is changed to *pull* as the grasp is considered achieved.

Follow action

This action is composed of two surgemes S_1^l and S_2^l. The system remains in the initial state S_1^l until the catheter becomes visible (trigger t_3^l), a condition that fires the state transition to S_2^l in which the left SARAS arm (SARAS2) follows the catheter within a predefined safe distance from the surgeon's tool currently grasping the catheter.

Grasp action

This action is composed of seven surgemes. The initial one (S_3^l) consists of a motion towards the catheter to signal the surgeon that the correct sequence of events towards the grasping action is occurring. When the tool reaches the desired position (trigger t_{14}^l), the next state is set to S_4^l to rotate the tool relative to the catheter's attitude. Once the desired rotation has been reached (t_8^l), the controller transits to the state S_5^l where the manipulator opens the grasper. At this point, when the grasper is successfully open (t_{12}^l), the next surgeme in S_6^l moves the tool to the final grasping point, which is computed as the current position of the da Vinci® tooltip, with a small offset to avoid tool collisions. Now, the system can transit to S_7^l whenever either the desired grasping point is reached (t_{14}^l) or the speech command "left arm close the instrument" (t_{21}^l) is requested. In this surgeme (S_7^l) the left arm closes the grasper and, once the tool has been totally closed (t_{10}^l), the next state is set to S_8^l. At this point, the tool is commanded back slightly along the RCM axis to carefully pull the catheter and thus verify whether the grasping was truly successful. When the desired position is reached (t_{14}^l), the system transits to the next state S_9^l which is a *no operations surgeme*.

Pull action

This action is composed of three surgemes, with the initial state S_{10}^l used to move the manipulator towards a point defined as the maximum traction point (from the pre-registered map). The transition to the next state S_{11}^l could happen if the desired position is reached (trigger t_{14}^l) or the "left arm stop" speech command (t_{18}^l) is provided by the main surgeon. The state S_{11}^l is a *no operations surgeme* and is intended to be the final surgeme of the procedure. If the speech command "left arm pull more" (t_{19}^l) is provided, the next state goes back to S_{10}^l; if the speech command is "left arm pull less" (t_{20}^l), the next state is set to S_{12}^l. This state moves the tool to a point nearby the bladder neck to release the catheter, and the next state is set to S_{11}^l if the desired position is reached or if the "left arm stop" sentence t_{18}^l is detected.

4.3.4 Right arm procedure

The procedure for the right arm is limited to three phases due to the scissors tool: these are the *idle*, the *bladder mobilization*, and the *vesicourethral anastomosis* phases. The *idle phase* is again used to keep the robot in a safe configuration and far from the main surgeon's tools. When the "right arm

push down the bladder" speech command is provided (trigger t^r_{13}) the system transits to the *bladder mobilization phase*. This phase is composed of three actions: *approach action*, *push action* and *release action*. The first action approaches the upper part of the bladder, while the second action pushes the bladder down and handles any possible adjustment of the applied pressure or contact point. The last action releases the bladder and moves back the manipulator to the safe position. The transition from the *approach action* and the *push action* happens when the end-effector reaches the top of the bladder (t^r_1). The transition from *push action* to *release action* occurs when the "right arm release" command is provided (t^r_{21}). When trigger t^r_{14} is fired, following the "right arm cut the needle" speech command, a phase change occurs towards the *vesicourethral anastomosis phase*. This phase, the last one of the procedure, is composed of a single action *cut action* consisting of moving the scissors to the thread and cut it.

Approach action

This action is composed of two surgemes S^r_1 and S^r_2. The initial state is S^r_1 where the end-effector is moved to the top of the bladder. In case the surgeon needs to stop the motion, the speech command "stop" (t^r_{22}) changes the current surgeme to S^r_2. In the state S^r_2 the robot is still and it keeps this state until the speech command "right arm retry" is provided (t^r_{22}).

Push action

This action is composed of four surgemes, where the initial state S^r_3 moves the robot to the bottom of the bladder increasing the pressure applied until either a "stop" speech command is received (t^r_{22}) or the bottom position is reached (t^r_{11}). In any case, the next state S^r_4 keeps the robot still and holds the bladder down. From this state three speech commands could fire a state transition: "right arm push more" (t^r_{18}) which sets the next state back to S^r_3; "right arm push less" (t^r_{19}) which let to transit to the state S^r_5 or "right arm push here" (t^r_{20}) which triggers the transition to the state S^r_6. The state S^r_5 moves the arm to the top of the bladder, decreasing the applied pressure until either a "stop" speech command is received (t^r_{22}) or the position is reached (t^r_{11}). The state S^r_6 moves the arm towards the main surgeon tool and is used to apply pressure to a different point. When the "stop" speech command is received (t^r_{22}) or the goal is reached (t^r_{11}), it transits back to S^r_4.

Release action

This action is composed of two surgemes S_7^r and S_8^r. The initial state S_7^r moves the tool back on the top of the bladder. When the desired position is reached (t_{11}^r), the state transits to S_8^r which moves the tool to the safe position.

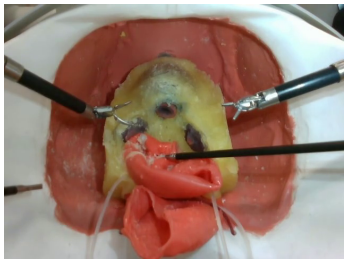
Cut action

This action is composed of five surgemes. The initial state S_9^r moves the robotic arm towards the position of the thread (which is in our case nearby the tip of the main surgeon's tool). When the desired thread position is reached (t_{11}^r), the system transits to S_{10}^r where the scissors are opened. When the tool is fully open, the trigger t_9^r is generated and the next state is set to S_{10}^r where the tool is rotated to be perpendicular to the thread. Once the desired rotation is reached (t_5^r), the state is set to S_{12}^r and the manipulator approaches the thread. In case the thread position is reached (t_{11}^r) or the speech command "right arm close the instrument" (t_{22}^r) is requested by the main surgeon, the next state is set to S_{13}^r and the robot cuts the thread. If the thread is missed, the surgeon can request by the command "right arm release" (t_{21}^r) to restart the action from S_9^r, otherwise the phase is completed.

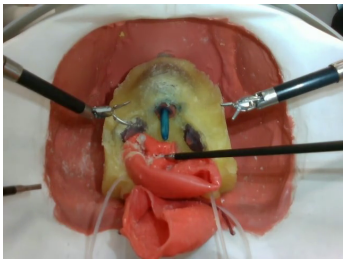
4.4 RARP execution and validation

To validate the proposed architecture, three surgeons with experience in robotic-assisted radical prostatectomy have been invited to execute the procedure in cooperation with the *SARAS* autonomous system. Before the validation, each surgeon has been briefed on the functionality of the system with an explanation on the speech commands list, the way the autonomous system detects the catheter, and the instructions on the user interface available to them in the da Vinci® console; the whole training took approximately twenty minutes. In total, the validation is composed of 38 executions of SARAS-related phases of the RARP, including 10 repetitions of the *bladder mobilization*, 14 of the *bladder neck transection*, and 14 of the *vesicourethral anastomosis*. In the following paragraphs, we proceed with an in-depth analysis of the bladder neck transection phase due to its comprehensive application of all the surgemes and its extensive interaction with the main surgeon. In Figure 4.4 a sequence of snapshots shows the main steps of the phase and in Figure 4.5 a sequence diagram shows the temporal relation between transitions.

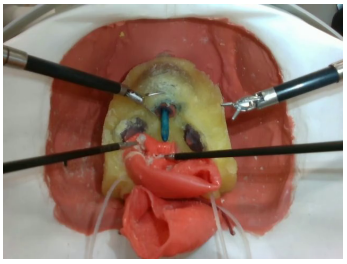
The procedure starts in the *idle* phase with the configuration shown in Figure 4.4a. The robot is still in a safe position waiting for the surgeon's speech



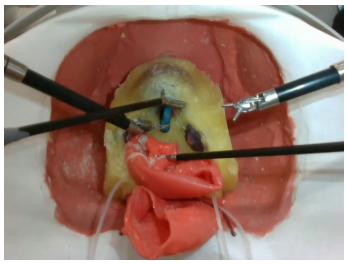
(a) Idle



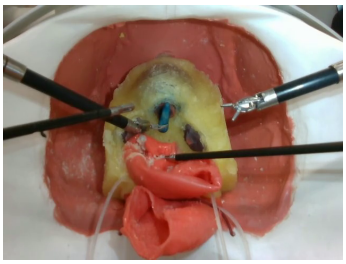
(b) Idle $\rightarrow$ bladder neck transection, follow, S_1^l



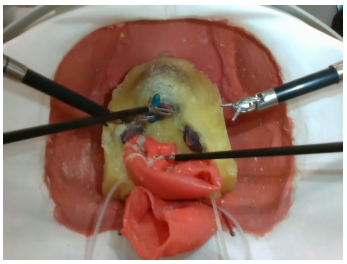
(c) $S_1^l \rightarrow S_2^l$



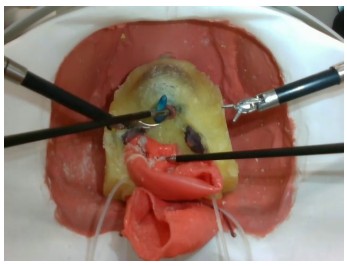
(d) Follow $\rightarrow$ grasp, $S_3^l \rightarrow S_4^l \rightarrow S_5^l \rightarrow S_6^l$



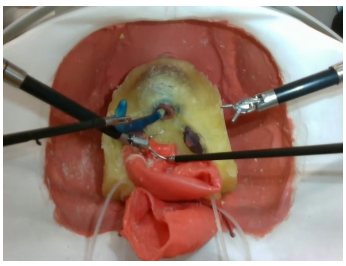
(e) $S_6^l \rightarrow S_7^l \rightarrow S_8^l \rightarrow S_9^l$



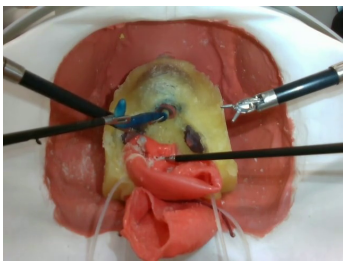
(f) Grasp $\rightarrow$ grasp, $S_3^l \rightarrow S_4^l \rightarrow S_5^l \rightarrow S_6^l$



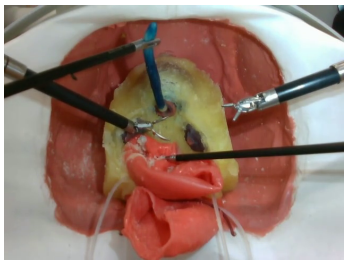
(g) $S_6^l \rightarrow S_7^l$



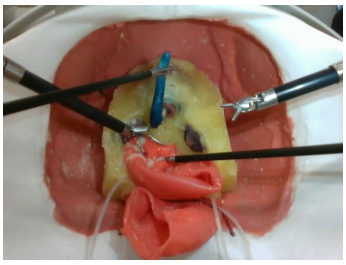
(h) $S_7^l \rightarrow S_8^l \rightarrow S_9^l$



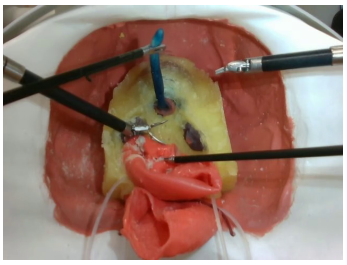
(i) S_9^l



(j) Grasp $\rightarrow$ pull, $S_{10}^l \rightarrow S_{11}^l$



(k) $S_{11}^l \rightarrow S_{12}^l \rightarrow S_{11}^l$



(l) $S_{11}^l \rightarrow S_{10}^l \rightarrow S_{11}^l$

Figure 4.4: A series of snapshots taken from the experimental validation during the *bladder neck transection*. State transitions are indicated in the captions as arrows and since some of them occurs quickly the transitions are sometimes grouped together in the same snapshot. A detailed view of the transitions occurrences is available in Figure 4.5.

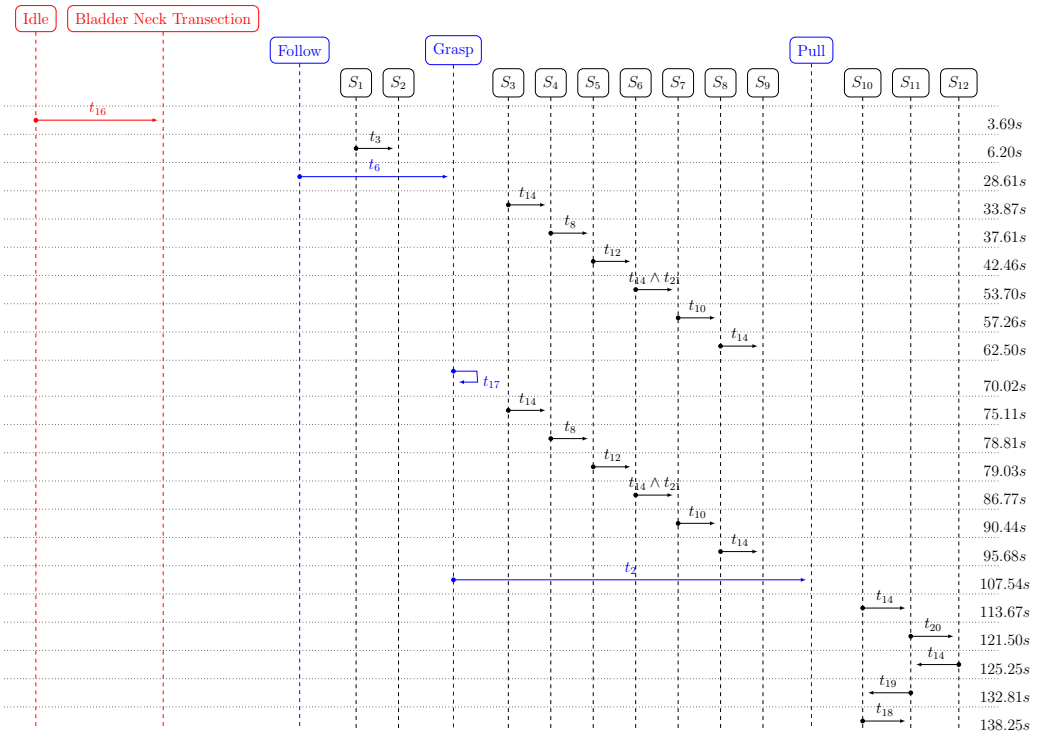
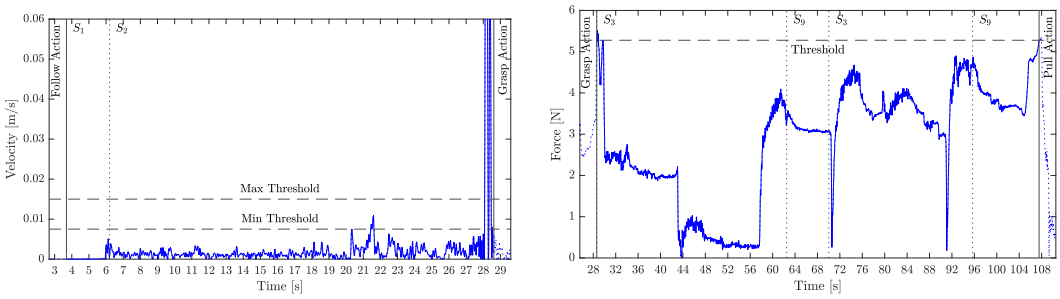


Figure 4.5: The sequence diagram shows the time when each transition occurred during the *bladder neck transection*. The time flow is represented by the vertical dashed lines. A transition that occurred between two states is indicated as an arrow, the label above the arrow indicates the trigger which activates the transition. On the right side of the diagram, the transition times are reported. The red, blue, and black colours are used to group the transitions occurring for phases, actions and surgemes respectively.



(a) Feature observer

(b) Force observer

Figure 4.6: The catheter speed and the modulus of the interaction force during the *follow* and *grasp* action respectively. a) the horizontal black dashed lines represent the *feature observer* threshold, the vertical black lines represent the action transition and the vertical dotted black lines represent the surgeme transitions, b) the horizontal black dashed line represents the *force observer* threshold, the vertical black lines represent the action transitions and the vertical dotted black lines represent the surgeme transitions.

command to trigger the *bladder neck transection* phase. When the command is received (t^l_{16}), the system transits to the *bladder neck transection* phase. As consequence also the current action and surgeme are modified, which are set respectively to $Follow$ and S^l_1, as shown in Figure 4.4b. In the meantime, the *catheter observer* is working concurrently, searching for the presence of the catheter.

Once the catheter is found (t^l_3) the current surgeme transits to S^l_2, and the autonomous system moves the end-effector at 3.5 cm apart from the catheter position as shown in Figure 4.4c. The main surgeon starts the catheter extraction by grasping and pulling upward the catheter with the intent of placing the catheter tip in a position that facilitates the grasping by the autonomous system. In the meantime, the *feature observer* analyzes the catheter velocity profile with respect to the threshold and action transitions. As shown in Figure 4.6a, the velocity exceeds the threshold for a sufficient time at 28.01 s and then drops to a value approximately close to 0 at 28.61 s. This behavior makes the *feature observer* to trigger t^l_6, thus the transition from *follow* to *grasp* is executed. Consequently, the current state of the surgeme is set to S^l_3, as shown in Figure 4.4d. The autonomous system moves the end-effectors of the left *SARAS* arm at 1 cm distance to the catheter (S^l_3); then it rotates the tool (S^l_4), opens the grasper (S^l_5), and moves again the tool at the grasping point (S^l_6) computed with respect to the current catheter position. The next transition guards are waiting for the triggers t^l_{14} and t^l_{21}: the former is given when the grasping point position is reached, while the latter is provided by the *speech observer* when the "left arm close the instrument" command is pro-

vided. When t^l_{14} and t^l_{21} are triggered, the surgeme S_7 closes the gripper, S^l_8 moves the end-effector to verify the grasping, which is followed by S^l_9 which puts the statechart in an idle state waiting for confirmation to continue. As shown in Figure 4.4e, during this attempt, the gripper missed the catheter. For this reason, the surgeon triggers the transition t^l_{17}, with the "left arm retry" speech command, as shown in Figure 4.4f, bringing back the current state to S^l_3. The execution is the same as before until the current state reaches S^l_6: this time the grasping is successful as shown in Figure 4.4g and the autonomous system reaches S^l_9 extracting the catheter as shown in Figure 4.4h. The surgeon pulls the catheter in order to generate a force on the autonomous arm which is monitored by the *force observer*. If the force reaches the desired threshold, the grasping is strong enough to proceed with the next phase of the procedure. Figure 4.6b shows the force profile.

Since the monitoring is performed by directly sensing the torques applied at the joints, these measurements could exceed the threshold multiple times during the motion execution due to the motor's controller action. For this reason, the transition could occur only when the surgeme state is S^l_9, which guarantees that the robot has already executed every step necessary to grasp the catheter and it is still. When the force exceeds the threshold (t^l_2), the current action moves to *Pull*, and the actual surgeme is set to S^l_{10}. Then the end-effector is moved to the pulling point and when the goal is reached the surgeme t^l_{14} switches the current state to S^l_{11} as shown in Figure 4.4j. The surgeon requests "left arm pull less" (t^l_{20}) with the specific speech command which sets the current state to S^l_{12} and the end-effector is moved near the bladder neck position as shown in Figure 4.4k. When the position is reached (t^l_{14}) the system gets back to the surgeme S^l_{11}. The catheter is retracted by the autonomous arm according to the main surgeon's instructions. At this point, the surgeon gives the speech commands to pull the catheter and to stop the motion when the end-effector reaches the desired position. This corresponds to the transition from S^l_{10} to S^l_{11}, and then back to S^l_{10} again. The catheter extraction is so concluded with success.

4.4.1 Execution evaluation and analysis

We have evaluated the developed system for autonomous RARP in terms of *phase success rate*, which is expressed as the number of succeeded phases on the total number of executions. We consider to be a successful phase whenever the surgeon is able to accomplish the goal without any fatal error (e.g., damages to the robots or the patient). Any time the surgeon has encountered minor issues

(e.g., missing the thread or the catheter grasping), but they has been able to recover the phase without any external intervention, the phase is considered successful. The first phase, the *bladder mobilization*, has minimal interaction with the surgeon, and its success is closely linked to the calibration of the robot with respect to the environment. For this reason, it has required a minimal amount of tests to accomplish the validation, with all of the performed tests being successful. No minor nor fatal issues have been encountered in testing this phase. The validation of the second phase, the *bladder neck transection*, in only one test out of 14 has encountered a fatal error which prevented the phase completion. The issue was caused by an erroneous value of the force threshold which let the statechart unable to transit on the trigger t_2 to the *Pull* action. This was caused by a technical mishap preventing the run-time threshold adaptation and the following procedure interruption. This phase also brought out several minor inconveniences, such as few collisions with the da Vinci® arms, minor collisions with the bladder during the movement of the SARAS 2 arm and misinterpreted speech commands promptly recovered by the surgeons' direct intervention. Finally, during the *vesicourethral anastomosis* phase validation, the system encountered a fatal error out of 14 executions. The issue was caused by the surgeon, since he/she failed to correctly point the desired needle approach position causing the SARAS1 arm to start moving toward an undesired and non-reachable configuration. This caused the robot to halt in a non-reachable position that led to a control system lock and a fatal error. Another issue we faced during this last phase was the SARAS 1 arm failure to accomplish the needle's thread cut. This can be caused by erroneous indications from the main surgeon mainly due to the low resolution of the images displayed on the da Vinci® console available with the dVRK platform. These failures, however, only led to the repetition of the positioning of the arm near the thread, but they did not prevent the phase completion.

Chapter 5

Improved passivity-based bilateral teleoperation

Bilateral teleoperation systems allow the telepresence of an operator while working remotely. Such ability becomes crucial when dealing with critical environments like R-MIS [98]. As mentioned in Chapter 4, in this thesis teleoperation plays an important role while handling unexpected events, i.e., when the assistant surgeon has to take over the *SARAS* arms.

The main properties of a teleoperation system are stability and transparency which, in general, are in contrast and they cannot be fully achieved at the same time. In the field of bilateral teleoperation a system is said to be transparent if the mechanical impedance felt by the user is the same as the impedance of the environment. The *two-layer* architecture is a passivity-based control technique able to guarantee the stability even in case of time delays in the communication channel and when the slave robot interacts with unknown environment. Such control methodology exploits the concepts of energy tanks [99], [100] and its main advantage is in the hierarchical subdivision between the transparency layer and the passivity layer. Energy tanks exploit the concept of passivity to build flexible and passivity-preserving interconnection. The larger the passivity margin, the more the system can absorb the energy generated by non-passive actions (e.g. changing the stiffness in a visco-elastic coupling) while preserving its passivity. The energy dissipated by the system is stored in a (virtual) energy tank and can be reused for implementing

Part of this chapter is based on the following publications:

- M. Minelli, N. Piccinelli, F. Falezza, F. Ferraguti, R. Muradore, and C. Secchi, *IEEE Transactions on Control Systems Technology*, 2022, Accepted
- F. Loschi, N. Piccinelli, D. Dall'Alba, R. Muradore, P. Fiorini, and C. Secchi, "An optimized two-layer approach for efficient and robustly stable bilateral teleoperation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 12 449–12 455

any desired control action in a passivity-preserving way.

In fact, the transparency layer computes the desired command and then the passivity layer modulates if needed the desired command to ensure a passive control action. This decomposition allows the implementation of different transparency policies without changing the passivity layer. Nonetheless, as we will see in Chapter 6 this solution can prevent an optimal control strategy.

In the following, we will present a novel theoretical formulation of the energy tank that improves the use of energy, allowing a consistent flow of energy between all the systems of the teleoperation architecture. Then, we will present an optimal modulation strategy to ensure a robustly stable behaviour for the system and the best passive implementation of the desired control input. This allows to get rid of all the heuristics which are in general adopted when the energy in the tank is not sufficient for implementing the desired control. All the results presented in this chapter assume constant communication delay. In general, the delay can be time-varying and some packets can get lost during the communication. Nonetheless, the results can be easily extended to passively dealing with these phenomena exploiting the strategy proposed in [101].

The validation of the methodologies proposed in this chapter has been performed on platforms different from the autonomous arms used in *SARAS*. Nonetheless, we plan to test such techniques with the Franka Emika arms soon.

5.1 State of the art

In [102] the transparency of the teleoperation in case of no communication delays is achieved using a coupling between the transmitted impedance and the impedance of the environment. The work of [103] introduced two new components in the teleoperation architecture: the passivity observer and the passivity controller. These components keep track of the energy entering or leaving the operator and environment sides and in case of instability dissipate the exceeding energy adding artificial damping. Passivity-based control strategies have been proven to be very successful since they allow to robustly handle the interaction with unstructured environments and to compensate the destabilizing effects of the communication delay. For example, the wave variables developed in [104]–[106] are one of the main tools used to design a stable teleoperation system and have been exploited for decades. Based on the scattering transformation, the wave variables encrypt the power variables (velocities and forces) exchanged between the local and the remote sides to turn the communication channel into a passive element, regardless of the time delays. Furthermore, if

both the local and remote sides are passive, the overall teleoperation architecture is passive too and thus stable. In [107] a PD-like coupling is used for robustly joining the local and the remote robots and in [108] a synchronization dynamics is implemented for achieving asymptotic position tracking. In these cases, a spurious force necessary for stabilization affects the force felt by the operator. The main drawback of passivity-based approaches is that passivity may hide the real interaction force that should be felt by the user.

Starting from the Time Domain Passivity Control (TDPC) algorithm developed in [109], its application to bilateral telemanipulation had been proposed in [110]. In their approach, two elements are introduced: the passivity observer (PO) and the passivity controller (PC). The PO monitors the energy flow into the system and a time-varying damping element, the PC, is activated to dissipate the excessive energy when necessary. An improved version of this kind of architecture, the Power-based Time-Domain Passivity Control (PTDPC), has been proposed in [111], where the power flow, rather than the energy flow, is monitored in order to achieve a smoother activation of the PC. Alternative approaches are based on the idea of predicting the non-delayed output of the plant by exploiting a model of the system, and so compensating the problem introduced by the delays. An example is the linear predictive controller called *Smith predictor* [112]. Unfortunately the linear approximation can degrade the effectiveness of the proposed method. In fact, in a teleoperation system, the local and the remote devices are haptic interfaces or robotic manipulators, which model is typically non-linear and may also vary with time (e.g., in case of user interaction or object picking).

In [113] a recurrent neural network has been introduced to model the remote robot non-linearity and taken into account it with a linear Smith predictor in order to improve the performance of the system. In [114] an adaptive control algorithm has been proposed to deal with time-varying environment dynamics. In [115] an online training of the network allows the system to estimate and map the remote device and environment dynamics at the local side. This increases the usability of the system, especially in the presence of substantial delays in the communication channel. Recently, several approaches trying to get rid of unwanted dynamic behaviours felt by the user have been proposed: from the use of variable dampers in [116], to the set-point modulation strategy in [117].

The *two-layer* architecture has been extended also to a multi-local multi-remote (MLMR) systems in [118] by introducing the concept of shared energy tank. A single tank is placed at each side of the teleoperation architecture and all the devices belonging to the same side share the energy inside that tank.

This allows thus to decrease the conservativeness introduced by the passivity preservation and to increase the transparency. Even though this architecture guarantees the stability of the system while allowing high level of flexibility and transparency, there are some drawbacks from the energy management point of view: energy is often wasted when the energy tank needs to be bounded.

If the energy stored in the tank is not sufficient for implementing the desired input, many application-dependent heuristics are exploited for guaranteeing an effective passivity-preserving implementation of the desired signal [119]–[121]. Recently, a different perspective on passivity has been addressed: passivity is considered as a constraint over which the behaviour to be implemented should be optimized [122], [123].

5.2 Novel energy tank dynamics

Let a gravity compensated robotic manipulator be modelled by the following n-DOFs Euler-Lagrange system

$$\boldsymbol{B}_w\left(\boldsymbol{x}_w(t)\right)\ddot{\boldsymbol{x}}_w(t)+\boldsymbol{C}_w\left(\boldsymbol{x}_w(t),\dot{\boldsymbol{x}}_w(t)\right)\dot{\boldsymbol{x}}_w(t)=\boldsymbol{F}_w^{\tau}(t)+\boldsymbol{F}_w^{ext}(t) \tag{5.1}$$

where $\boldsymbol{x}_w(t)\in\mathbb{R}^n$ are the coordinates of the configuration of the end-effector in the task space with $w\in\{l,r\}$. The subscripts l and r are used to indicate the local and the remote side, respectively. The term $\boldsymbol{B}_w\left(\boldsymbol{x}_w(t)\right)\in\mathbb{R}^{n\times n}$ is the symmetric and positive-definite inertia matrix, $\boldsymbol{C}_w\left(\boldsymbol{x}_w(t),\dot{\boldsymbol{x}}_w(t)\right)\in\mathbb{R}^{n\times n}$ is the Coriolis/centrifugal matrix. The term $\boldsymbol{F}_w^{\tau}(t)\in\mathbb{R}^n$ represents the control inputs vector while $\boldsymbol{F}_w^{ext}(t)\in\mathbb{R}^n$ is the vector of generalized external forces (i.e., the force applied by the user or the force applied on the environment). For ease of notation we consider that all robots have the same number of DOFs. All the results can be easily generalized to the case where the robots have a different number of DOFs. With a slight abuse of notation, we will hereafter use $\boldsymbol{B}_w(t)$ and $\boldsymbol{C}_w(t)$ to indicate, respectively, the value of $\boldsymbol{B}_w\left(\boldsymbol{x}_w(t)\right)$ and $\boldsymbol{C}_w\left(\dot{\boldsymbol{x}}_w(t),\boldsymbol{x}_w(t)\right)$ at time t. Moreover, for clarity of presentation, we will omit the time-dependence of the variables when the context is clear.

In order to be able to fill the tank when necessary, a controlled dissipation is implemented on each robot, and the dissipated energy flows into the tank. This can be done by splitting the control input of each robot into the sum of two terms

$$\boldsymbol{F}_w^{\tau}(t)=\boldsymbol{\phi}_w^d(t)+\boldsymbol{\phi}_w^t(t). \tag{5.2}$$

The first term implements a variable local damping

$$\boldsymbol{\phi}_w^d(t)=-\boldsymbol{D}_w(t)\dot{\boldsymbol{x}}_w(t) \tag{5.3}$$

where $\boldsymbol{D}_w(t) \in \mathbb{R}^{n\times n}$ is a time-varying positive semi-definite matrix. The second term $\boldsymbol{\phi}_w^t(t)$ is the control input. By embedding the damping injection into (5.1) we get the following damped Euler-Lagrange model for each robot

$$\boldsymbol{B}_w\ddot{\boldsymbol{x}}_w + \boldsymbol{C}_w\dot{\boldsymbol{x}}_w + \boldsymbol{D}_w\dot{\boldsymbol{x}}_w = \boldsymbol{\phi}_w^t + \boldsymbol{F}_w^{ext} \tag{5.4}$$

An energy tank needs to be placed at each side of the teleoperation system. With respect to the formulation of the shared energy tank in [118] we also propose the following formulation

$$\begin{cases} \dot{x}_{t_w} = \sigma_w \left(\dfrac{\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w}{x_{t_w}} + u_{t_w} \right) \\ y_{t_w} = \dfrac{\partial T_w}{\partial x_{t_w}} \end{cases} \tag{5.5}$$

where $x_{t_w} \in \mathbb{R}$ is the state of the tank, $(u_{t_w}, y_{t_w}) \in \mathbb{R}^2$ is the power port through which the tank can exchange energy with the rest of the world, and

$$T_w(x_{t_w}) = \frac{1}{2}x_{t_w}^2 \tag{5.6}$$

is the energy stored in the tank. Each robot is interconnected to the energy tank to use the stored energy to execute the desired action. This can be done by implementing the following power preserving interconnection between the robots and the energy tanks

$$\begin{cases} \boldsymbol{\phi}_w^t = \boldsymbol{\omega}_w y_{t_w} \\ u_{t_w} = -\boldsymbol{\omega}_w^T \dot{\boldsymbol{x}}_w \end{cases} \tag{5.7}$$

such that

$$\dot{\boldsymbol{x}}_w^T \boldsymbol{\phi}_w^t = -u_{t_w} y_{t_w} \tag{5.8}$$

which means that each robot can extract/inject energy from/into the tank. The whole teleoperation system is then modelled by

$$\begin{cases} \boldsymbol{B}_w\ddot{\boldsymbol{x}}_w + \boldsymbol{C}_w\dot{\boldsymbol{x}}_w + \boldsymbol{D}_w\dot{\boldsymbol{x}}_w = \boldsymbol{\omega}_w x_{t_w} + \boldsymbol{F}_w^{ext} & \text{(5.9a)} \\ \dot{x}_{t_w} = \sigma_w \left(\dfrac{\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w}{x_{t_w}} - \boldsymbol{\omega}_w^T \dot{\boldsymbol{x}}_w \right). & \text{(5.9b)} \end{cases}$$

The term $\sigma_w \in \{0, 1\}$ is used to limit the energy stored into the tank, because as described in [117], it is necessary not to store too much energy in the tank in order to avoid the implementation of practically unstable behaviours. In [118] the upper bound of the energy stored in the tank was defined as

$$\sigma_w = \begin{cases} 0 & \text{if } T_w(x_{t_w}) = T_w^{max} \\ 1, & \text{otherwise} \end{cases} \tag{5.10}$$

causing a wasting of energy. Here, we propose to design a novel energy upper bounding strategy where the whole excess of energy can be reused, decreasing the conservativeness of the overall system. In [118], when the tank is full, σ_w is used to bound only the dissipation term

$$\frac{\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w}{x_{t_w}} \tag{5.11}$$

while the energy introduced by the controller $\boldsymbol{\omega}_w^T \dot{\boldsymbol{x}}_w$ is used or wasted if the energy is extracted or injected, respectively. We aim to develop a less conservative system allowing the excess of energy to be reused within the architecture. We re-define the terms σ_w as follows:

$$\sigma_w = \begin{cases} 0, & \text{if } T_w(x_{t_w}) = T_w^{max} \wedge \dfrac{\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w}{x_{t_w}} - \boldsymbol{\omega}_w^T \dot{\boldsymbol{x}}_m > 0 \\ 1, & \text{otherwise} \end{cases} \tag{5.12}$$

where T_w^{max} represents the energy upper bound. The novel shared energy tank allows to bound the maximum level of energy stored into the tank by energetically disconnecting the tank from the robots. It is worth noting that the disconnection occurs only when $\sigma_w = 0$, that is

$$T_w(x_{t_w}) = T_w^{max} \tag{5.13}$$

and

$$\frac{\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w}{x_{t_w}} - \boldsymbol{\omega}_w^T \dot{\boldsymbol{x}}_m > 0 \tag{5.14}$$

This means that the energy tank is full (5.13) and the overall control action is dissipative (5.14) (i.e., injecting energy into the tank), and so it is safe to decouple the tank and the robot. In this situation, the energetic disconnection between the tank and the robots allows to keep the energy stored into the tank constant to the maximum value T_w^{max}, since no more energy can flow to the tank.

When $\sigma_w = 0$, the extra energy would produce an evolution of $\dot{x}_{t_w}$ represented by the term

$$\frac{\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w}{x_{t_w}} - \boldsymbol{\omega}_w^T \dot{\boldsymbol{x}}_w \tag{5.15}$$

Since $\sigma_w = 0$, this extra energy does not produce any effects on the tank level and it can not be reused by the system. Nevertheless, in a multiple tanks system, this extra energy can be sent to the tank at the other side of the network, i.e., from the local side to the remote side or vice-versa. This formulation of the shared energy tank will be implemented and analysed in the next section where the overall teleoperation architecture is considered. Another

problem related to the level of energy stored into the tank is its minimum value: if $x_{t_w} = 0$, i.e., no energy left in the tank, the (5.9) becomes singular. To avoid this problem we initialize the tank with some energy $T_w^{min} > 0$ and we prevent excessive energy extraction by designing the value of the local damping of each robot

$$D_w = \begin{cases} D_w^{min}, & \text{if} \quad T_w(x_{t_w}) > T_w^{b_{max}} \\ \xi\big(T_w(x_{t_w})\big), & \text{if} \quad T_w^{b_{min}} \leq T_w(x_{t_w}) \leq T_w^{b_{max}} \\ D_w^{max}, & \text{if} \quad T_w(x_{t_w}) < T_w^{b_{min}} \end{cases} \tag{5.16}$$

as shown in [118]. If the energy in the tank exceeds the energy upper threshold $T_w^{b_{max}}$ the local damping injection of the robot is set to a minimum level D_w^{min} since harvesting energy is not needed. If the energy in the tank is going below the energy lower threshold $T_w^{b_{min}}$, the local damping injection of the robot is set to a maximum level D_w^{max}, in order to fill energy into the tank as quickly as possible. Otherwise $D_w = \xi(T_w(x_{t_w}))$, where $\xi(T) : \mathbb{R} \to \mathbb{R}^{n\times n}$ is any smooth non-increasing function such that $D_W = D_w^{min}$ if $T_w(x_{t_w}) = T_w^{b_{max}}$ and $D_w = D_w^{max}$ if $T_w(x_{t_w}) = T_w^{b_{min}}$. The choice of $\xi(T_w(x_{t_w}))$ guarantees a smooth transition between D_w^{min} and D_w^{max}, without discontinuities in the forces applied to the devices.

The modulation factor $\boldsymbol{\omega}_w$ can be set as in [118], [124] such that if the energy stored in the tank is less than a user-defined threshold a scaled version of the desired input is implemented. In the worst case such modulation must implies that no commands will be implemented in order to preserve passivity.

Proposition 1. *The system in* (5.9) *is passive with respect to the pair* $(\boldsymbol{F}_w^{ext}, \dot{\boldsymbol{x}}_w)$.

Proof. The kinetic energy $V_w(t)$ of the system in (5.9a) is given by the sum of the kinetic energies of all the robots at the w side and can be defined as

$$V_w(t) = \frac{1}{2}\dot{\boldsymbol{x}}_w(t)^T \boldsymbol{B}_w(t)\dot{\boldsymbol{x}}_w(t) \tag{5.17}$$

while $\dot{V}_w(t)$ is given by

$$\dot{V}_w(t) = \dot{\boldsymbol{x}}_w^T(t)\boldsymbol{B}_w(t)\ddot{\boldsymbol{x}}_w(t) + \frac{1}{2}\dot{\boldsymbol{x}}_w^T(t)\dot{\boldsymbol{B}}_w(t)\dot{\boldsymbol{x}}_w(t) \tag{5.18}$$

Using (5.9a) we can rewrite (5.18) as

$$\dot{V}_w(t) = \dot{\boldsymbol{x}}_w^T\boldsymbol{\omega}_w x_{t_w} + \dot{\boldsymbol{x}}_w^T\boldsymbol{F}_w^{ext} - \dot{\boldsymbol{x}}_w^T\boldsymbol{D}_w\dot{\boldsymbol{x}}_w + \dot{\boldsymbol{x}}_w^T(\dot{\boldsymbol{B}}_w(t) - 2\boldsymbol{C}_w(t))\boldsymbol{x}_w(t) \tag{5.19}$$

and since $\dot{\boldsymbol{B}}_w(t) - 2\boldsymbol{C}_w(t)$ is skew symmetric, then the last term disappear and thus we can further rewrite (5.18) as

$$\dot{V}_w(t) = \dot{\boldsymbol{x}}_w^T\boldsymbol{\omega}_w x_{t_w} + \dot{\boldsymbol{x}}_w^T\boldsymbol{F}_w^{ext} - \dot{\boldsymbol{x}}_w^T\boldsymbol{D}_w\dot{\boldsymbol{x}}_w \tag{5.20}$$

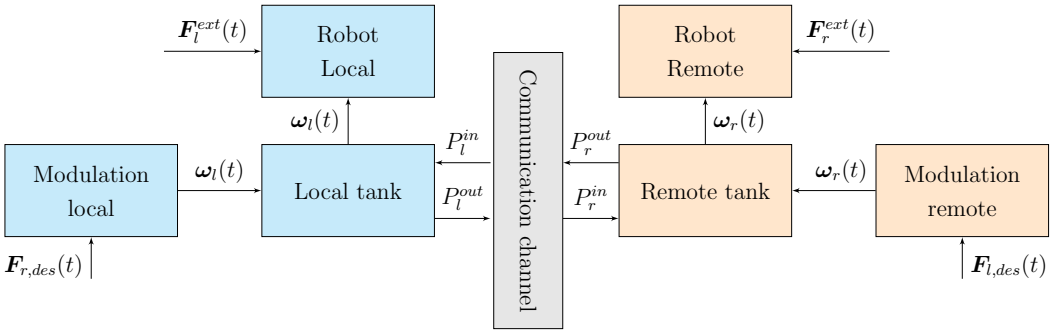


Figure 5.1: The control scheme of the bilateral teleoperation with the novel tank dynamic.

Substituting (5.9b) in (5.6) we can write $\dot{T}_w(t)$ as

$$\dot{T}_w(t) = \sigma_w(\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w - x_{t_w} \boldsymbol{\omega}_w^T \dot{\boldsymbol{x}}_w) \tag{5.21}$$

Now, consider as a storage function the total energy of the teleoperation system (5.9)

$$\mathcal{V}(t) = V_w(t) + T_w(t) \tag{5.22}$$

where $V_w(t)$ represents the energy associated to the local or remote side and T_w the energy stored in the corresponding tank. From (5.22) it follows that

$$\dot{\mathcal{V}}(t) = \dot{V}_w(t) + \dot{T}_w(t) \tag{5.23}$$

Substituting (5.20) and (5.21) in (5.23) we obtain

$$\dot{\mathcal{V}}(t) = \dot{\boldsymbol{x}}_w^T \boldsymbol{F}_w^{ext} - (1 - \sigma_w)(\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w - x_{t_w} \boldsymbol{\omega}_w^T \dot{\boldsymbol{x}}_w) \tag{5.24}$$

From (5.12), $\sigma_w \in \{0, 1\}$ and $\sigma_w = 0$ if and only if $\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w - x_{t_w} \boldsymbol{\omega}_w^T \dot{\boldsymbol{x}}_w \geq 0$. Thus, it follows that

$$\dot{\mathcal{V}}(t) \leq \dot{\boldsymbol{x}}_w^T \boldsymbol{F}_w^{ext} \tag{5.25}$$

which implies the passivity condition

$$\mathcal{V}(t) - \mathcal{V}(0) \leq \int_0^t \dot{\boldsymbol{x}}_w^T(\tau) \boldsymbol{F}_w^{ext}(\tau) d\tau \tag{5.26}$$

□

Since the passivity condition takes into account both the velocity and force of the robotic system, the proposed method can be used either with admittance or impedance causality.

5.2.1 The bilateral teleoperation

Following the approach reported in [100], we endow each shared energy tank with two ports, P_w^{out} and P_w^{in}, through which the tank can send/receive extra power to/from the rest of the world

$$\begin{cases} \dot{x}_{t_w} = \frac{\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w + \sigma_w P_w^{in}(t)}{x_{t_w}} - \frac{P_w^{out}(t)}{x_{t_w}} + u_{t_w} \\ y_{t_w} = \dfrac{\partial T_w}{\partial x_{t_w}} \end{cases} \tag{5.27}$$

These ports allow the interconnection of the local and the remote sides by means of a delayed communication channel. The general architecture can be decomposed into two layers: a *Transparency Layer* and a *Passivity Layer*. In the Transparency Layer, local and remote sides exchange the corresponding position, velocity and force measurements that are used for computing the desired control inputs ($\boldsymbol{F}_l^d$, $\boldsymbol{F}_r^d$). These forces are sent to the Passivity Layer which role is to passively implement them using the energy stored in the tanks. The overall architecture, shown in Figure 5.1, can be modelled as

$$\begin{cases} \boldsymbol{B}_l \ddot{\boldsymbol{x}}_l + \boldsymbol{C}_l \dot{\boldsymbol{x}}_l + \boldsymbol{D}_l \dot{\boldsymbol{x}}_l = \boldsymbol{\omega}_l x_{t_l} + \boldsymbol{F}_l^{ext} \\ \dot{x}_{t_l} = \dfrac{\dot{\boldsymbol{x}}_l^T \boldsymbol{D}_l \dot{\boldsymbol{x}}_l + \sigma_l P_l^{in}}{x_{t_l}} - \boldsymbol{\omega}_l^T \dot{\boldsymbol{x}}_l - \dfrac{P_l^{out}}{x_{t_l}} \\ \boldsymbol{B}_r \ddot{\boldsymbol{x}}_r + \boldsymbol{C}_r \dot{\boldsymbol{x}}_r + \boldsymbol{D}_r \dot{\boldsymbol{x}}_r = \boldsymbol{\omega}_r x_{t_r} + \boldsymbol{F}_r^{ext} \\ \dot{x}_{t_r} = \dfrac{\dot{\boldsymbol{x}}_r^T \boldsymbol{D}_r \dot{\boldsymbol{x}}_r + \sigma_r P_r^{in}}{x_{t_r}} - \boldsymbol{\omega}_r^T \dot{\boldsymbol{x}}_r - \dfrac{P_r^{out}}{x_{t_r}} \end{cases} \tag{5.28}$$

where $P_l^{in}, P_r^{in} \geq 0$ and $P_l^{out}, P_r^{out} \geq 0$ are incoming and outgoing power flows that the tanks can exchange with each other by means of the communication channel for balancing the two levels. The interconnection between the two sides of the teleoperation system can be represented by

$$\begin{cases} P_r^{in}(t) = P_l^{out}(t-\delta) \\ P_l^{in}(t) = P_r^{out}(t-\delta) \end{cases} \tag{5.29}$$

where δ represents the communication delay between the two sides. The policy used to define P_l^{out} and P_r^{out} in (5.28) is

$$\begin{cases} P_l^{out}(t) = (1-\sigma_l)(\dot{\boldsymbol{x}}_l^T \boldsymbol{D}_l \dot{\boldsymbol{x}}_l - x_{t_l} \boldsymbol{\omega}_l^T \dot{\boldsymbol{x}}_l) + E_r^{req}(t-\delta)\beta_l \bar{P} \\ P_r^{out}(t) = (1-\sigma_r)(\dot{\boldsymbol{x}}_r^T \boldsymbol{D}_r \dot{\boldsymbol{x}}_r - x_{t_r} \boldsymbol{\omega}_r^T \dot{\boldsymbol{x}}_r) + E_l^{req}(t-\delta)\beta_r \bar{P} \end{cases} \tag{5.30}$$

where $\bar{P} \in \mathbb{R}_0^+$ is a design parameter which represents the rate of energy flowing from one tank to the other, and the flags E_l^{req}, E_r^{req} are used to implement an

energy request process, and are defined as

$$E_w^{req} = \begin{cases} 1, & \text{if } T_w(x_{t_w}) \leq T_w^{req} \\ 0, & \text{otherwise} \end{cases}. \tag{5.31}$$

If the energy stored in the tank is under the user-defined threshold $T_w^{req} \in \mathbb{R}$ then the tank send an energy request signal E_w^{req} to the other tank, which can provide energy under the following condition

$$\beta_w = \begin{cases} 1, & \text{if } T_w(x_{t_w}) \geq T_w^{ava} \\ 0, & \text{otherwise} \end{cases} \tag{5.32}$$

namely, each tank can provide energy to the other side if the energy stored is larger than the user-defined threshold $T_w^{ava} \in \mathbb{R}$. The definition of the energy tank is slightly different with respect to the one in (5.9). This is due to the fact that the introduction of the signals P_l^{in}, P_r^{in}, and P_l^{out}, P_r^{out} allow to share the power provided by the controller between the tanks, increasing the promptness of the overall system. Since the behaviour of σ_l and σ_r are still defined by (5.12), in case $\sigma_l = 0$, from (5.28) it results

$$\dot{x}_{t_l} = \frac{\dot{\boldsymbol{x}}_l^T \boldsymbol{D}_l \dot{\boldsymbol{x}}_l}{x_{t_l}} - \boldsymbol{\omega}_l^T \dot{\boldsymbol{x}}_l - \frac{P_l^{out}}{x_{t_l}} \tag{5.33}$$

Substituting (5.30) in (5.33) it follows that

$$\dot{x}_{t_l} = -\frac{E_r^{req}(t-\delta)\beta_l \bar{P}}{x_{t_l}} \tag{5.34}$$

which means that the energy in the tank can only decrease and the power dissipated by the damping and the operator action is transferred through P_l^{out} to the other side. The input power is cancelled out in order to avoid energy storing in the communication channel. Otherwise, if $\sigma_l = 1$, from (5.28) it results that

$$\dot{x}_{t_l} = \frac{\dot{\boldsymbol{x}}_l^T \boldsymbol{D}_l \dot{\boldsymbol{x}}_l + P_l^{in}}{x_{t_l}} - \boldsymbol{\omega}_l^T \dot{\boldsymbol{x}}_l - \frac{E_r^{req}(t-\delta)\beta_l \bar{P}}{x_{t_l}} \tag{5.35}$$

which means that the tank doesn't need to be upper-bounded and all the power sources can modify its level. The same behaviour holds for the remote side. The strategy illustrated so far guarantees the passivity of the teleoperation system as we will prove. At first, we need to guarantee that the new definition of $P_w^{out}(t)$ still satisfies the condition described in [118], as the following lemma states.

Lemma 1. *$P_w^{out}(t) \geq 0$ with $w \in \{l, r\}$.*

Proof. Since $\sigma_w, E_w^{req}, \beta_w \in \{0,1\}$ and $\bar{P} \geq 0$ we get

$$E_w^{req}(t-\delta)\beta_w \bar{P} \geq 0 \tag{5.36}$$

Thus, from (5.30) it follows that $P_w^{out}(t)$ is positive if and only if

$$(1-\sigma_w)(\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w - x_{t_w} \boldsymbol{\omega}_w^T \dot{\boldsymbol{x}}_w) \geq -E_w^{req}(t-\delta)\beta_w \bar{P} \tag{5.37}$$

If $\sigma_w = 1$, (5.37) becomes

$$-E_w^{req}(t-\delta)\beta_w \bar{P} \leq 0 \tag{5.38}$$

that is always true thanks to (5.36). If $\sigma_w = 0$, from (5.12) we have

$$(\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w - x_{t_w} \boldsymbol{\omega}_w^T \dot{\boldsymbol{x}}_w) \geq 0 \tag{5.39}$$

which satisfy (5.37). This is the only case where the input term contributes to $P_w^{out}(t)$. As a consequence, $P_w^{out} \geq 0$ and this proves the lemma. □

We are now ready to state the main result.

Proposition 2. *The teleoperation system in* (5.28) *is passive with respect to the pair* $((\boldsymbol{F}_l^{ext}, \boldsymbol{F}_r^{ext}), (\dot{\boldsymbol{x}}_l, \dot{\boldsymbol{x}}_r))$.

Proof. We consider as storage function the total energy of the teleoperation system

$$W(t) = V_l(t) + V_r(t) + T_l(t) + T_r(t) + H_{ch}(t) \tag{5.40}$$

where $H_{ch}(t)$ is the energy stored in the communication channel. Using (5.28) we have that

$$\begin{aligned}\dot{W}(t) = {} & \dot{\boldsymbol{x}}_l^T \boldsymbol{F}_l^{ext} - \dot{\boldsymbol{x}}_l^T \boldsymbol{D}_l \dot{\boldsymbol{x}}_l + \dot{\boldsymbol{x}}_l^T \boldsymbol{\omega}_l x_{t_l} + \dot{\boldsymbol{x}}_r^T \boldsymbol{F}_r^{ext} - \dot{\boldsymbol{x}}_r^T \boldsymbol{D}_r \dot{\boldsymbol{x}}_r + \dot{\boldsymbol{x}}_r^T \boldsymbol{\omega}_r x_{t_r} + \\ & + \dot{\boldsymbol{x}}_l^T \boldsymbol{D}_l \dot{\boldsymbol{x}}_l - x_{t_l} \boldsymbol{\omega}_l^T \dot{\boldsymbol{x}}_l + \sigma_l P_l^{in}(t) - P_l^{out}(t) + \dot{\boldsymbol{x}}_r^T \boldsymbol{D}_r \dot{\boldsymbol{x}}_r - x_{t_r} \boldsymbol{\omega}_r^T \dot{\boldsymbol{x}}_r + \\ & + \sigma_r P_r^{in}(t) - P_r^{out}(t) + \dot{H}_{ch}(t)\end{aligned} \tag{5.41}$$

While the power is traveling from one side to the other, it is stored in the communication channel that becomes an energy storing element in the teleoperation system. In particular, as shown in [119], we have that

$$H_{ch}(t) = \int_{t-\delta}^{t} (P_l^{out}(\tau) + P_r^{out}(\tau)) d\tau \tag{5.42}$$

and by taking the time derivate

$$\dot{H}_{ch}(t) = P_l^{out}(t) - P_l^{out}(t-\delta) + P_r^{out}(t) - P_r^{out}(t-\delta) \tag{5.43}$$

Considering (5.29) and replacing (5.43) in (5.70) we get

$$\dot{W}(t) = \dot{\boldsymbol{x}}_l^T \boldsymbol{F}_l^{ext} + \dot{\boldsymbol{x}}_r^T \boldsymbol{F}_r^{ext} - (1-\sigma_l)P_l^{in}(t) - (1-\sigma_r)P_r^{in}(t) \tag{5.44}$$

Since $\sigma_w \in \{0,1\}$ and from Lemma 1 $P_w^{in} \geq 0$, it follows that

$$\dot{W}(t) \leq \dot{\boldsymbol{x}}_l^T \boldsymbol{F}_l^{ext} + \dot{\boldsymbol{x}}_r^T \boldsymbol{F}_r^{ext} \tag{5.45}$$

which implies the passivity condition

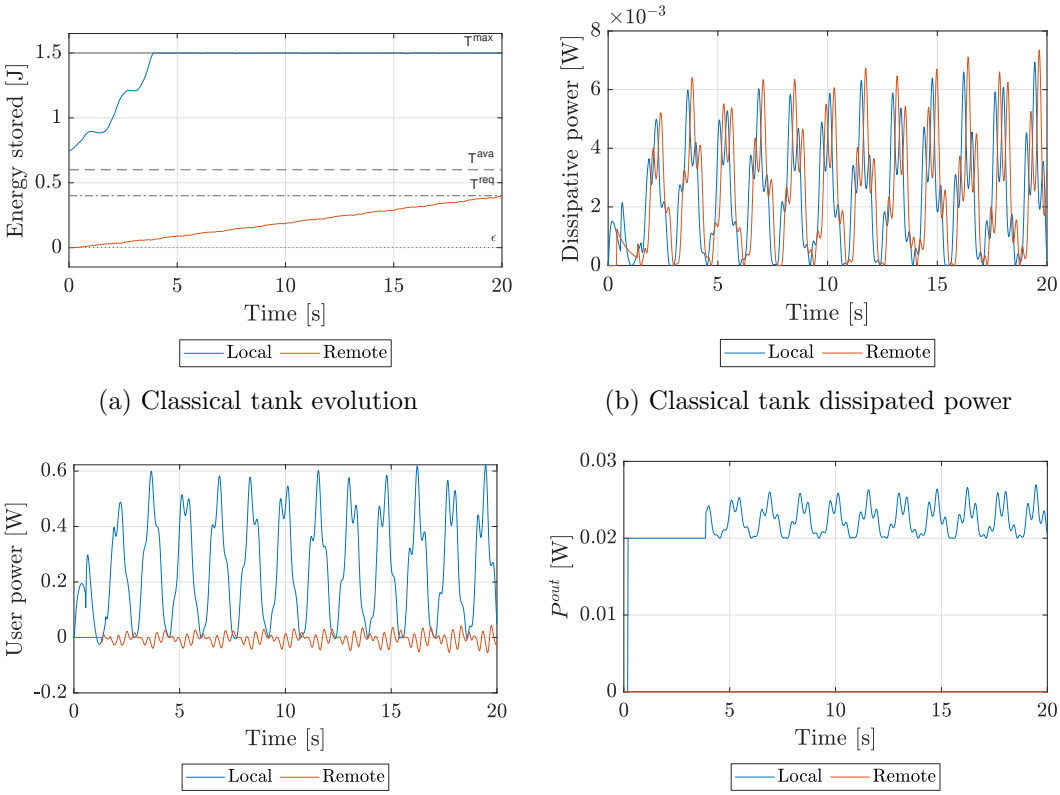
$$W(t) - W(0) \leq \int_0^t \dot{\boldsymbol{x}}_l^T(\tau)\boldsymbol{F}_l^{ext}(\tau) + \dot{\boldsymbol{x}}_r^T(\tau)\boldsymbol{F}_r^{ext}(\tau)d\tau \tag{5.46}$$

□

5.2.2 Simulation comparison

Due to the difficulty of evaluating the differences between the novel tank dynamics (5.5) proposed and its classical definition [118], we perform the comparison in a simulated environment. We also decide to simplify the robot model and use two rotational 1-DOF robotic manipulators (modelled as second-order dynamic systems) since the behaviour of the tank does not depend on the number of DOFs. The interaction torque provided by the operator is modelled as a position PD controller where the inputs are the reference signal and the delayed position and velocity of the robot at the remote side. This feedback loop models the classic video streaming used in a real teleoperation setup, where the operator uses the perceived remote side robot position to move the haptic device. The transparency layer is built with two position PD controllers, while the passivity layer modulates the desired command if and only if the corresponding tank has not enough energy. The simulation environment has been developed using *Simulink* on *Matlab R2022a*.

The test is conducted with a constant delay of 150 ms between the operator and the environment side. The sampling time of the controller was set to $\Delta t = 20\,\text{ms}$. The energy thresholds $T^{max} = 1.5$, $T^{ava} = 0.6$, $T^{req} = 0.4$, $\epsilon = 0.001$, $\bar{P} = 0.01$ are set to such values to force the local and remote side tanks towards the upper and the lower bound respectively. The tanks are initialised to $T^{max}/2$ at the local side and to ϵ at the remote side. The robot model parameters (i.e., inertia and friction) are set equal to $j = 0.0266\,kg\,m^2$ and $b = 0.0218\,N\,s$, respectively. As shown in Figure 5.2a and Figure 5.3a the tanks at the remote side are almost empty. The behaviours of the tanks between 0 sec to 5 sec are the same since the increase of the stored energy is only driven by the energy sent by means of $\bar{P}$ from the local to the remote side.



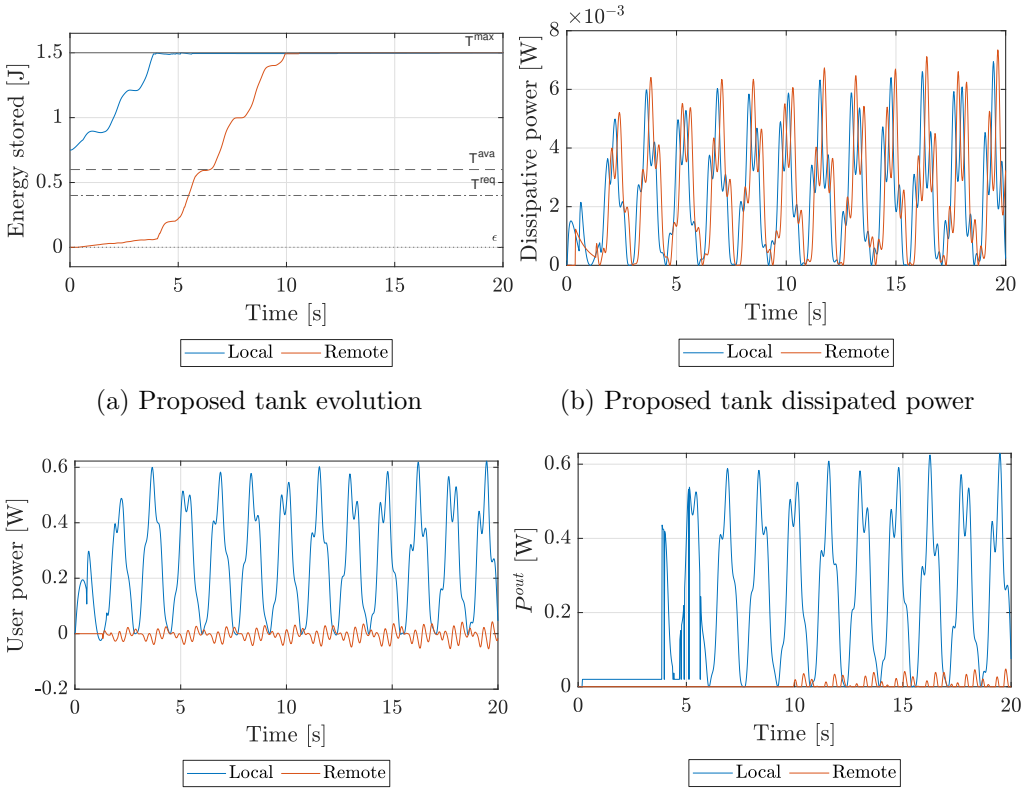
(a) Classical tank evolution

(b) Classical tank dissipated power

(c) Classical tank input power

(d) Classical tank output power

Figure 5.2: Time evolution of the classical tank dynamics. In red and blue the local and remote side tank respectively. a) the evolution of the tank, b) the power dissipated by the robots, c) the power generated by the control action, d) the output power.



(a) Proposed tank evolution

(b) Proposed tank dissipated power

(c) Proposed tank input power

(d) Proposed tank output power

Figure 5.3: Time evolution of the novel tank dynamics. In red and blue the local and remote side tank respectively. a) the evolution of the tank, b) the power dissipated by the robots, c) the power generated by the control action, d) the output power.

In both cases once reached 5 sec, the teleoperated robots have enough energy to perform their control action and start following the same trajectory. For this reason, as shown in Figures 5.2b, 5.2c, 5.3b, 5.3b, the dissipated power and the power provided by the controller in the transparency layer are the same in both cases.

The effect of the proposed novel tank dynamics can be clearly seen comparing Figure 5.2a with Figure 5.3a. With the classical tank definition, when the tank at the local side reaches its upper bound it starts to provide the dissipated power and the energy quantum $\bar{P}$ to the remote side tank. This results in a linear increase in the remote side tank until the E^{req} has been reached, around 20 sec. On the other hand, our tank dynamic adds to the dissipated power and the energy quantum $\bar{P}$, also the power dissipated by the control action. In fact, as shown in Figure 5.3d, the power shared by the tank at the local side starting from 5 sec includes the input power shown in Figure 5.3c. This additional power source, see Figure 5.3a, increases the tank level at the remote side reaching the upper bound around 9 s. The proposed solution is able to improve the promptness of the whole teleoperation system and to decrease the conservativeness related to the passivity approach.

5.3 Optimal modulation for passivity layer

As in the previous section, consider a system composed of a local and a remote manipulator, interacting with the human operator and the environment, respectively. Each robot, is fully actuated and gravity compensated and modelled as an n-DOFs Euler-Lagrange system

$$\boldsymbol{B}_w\left(\boldsymbol{x}_w(t)\right)\ddot{\boldsymbol{x}}_w(t)+\boldsymbol{C}_w\left(\boldsymbol{x}_w(t),\dot{\boldsymbol{x}}_w(t)\right)\dot{\boldsymbol{x}}_w(t)+\boldsymbol{D}\dot{\boldsymbol{x}}_w(t)=\boldsymbol{F}_w(t)+\boldsymbol{F}_w^{ext}(t) \tag{5.47}$$

where $\boldsymbol{x}_w(t),\dot{\boldsymbol{x}}_w(t)\in\mathbb{R}^n$ are pose and velocity of the end-effector. The matrices $\boldsymbol{B}_w\left(\boldsymbol{x}_w(t)\right),\boldsymbol{C}_w\left(\boldsymbol{x}_w(t)\right),\dot{\boldsymbol{x}}_w(t),\boldsymbol{D}(x_w(t))\in\mathbb{R}^{n\times n}$ are the positive definite inertia matrix, the Coriolis/centrifugal matrix and the damping matrix, respectively. The term $\boldsymbol{F}_w(t)\in\mathbb{R}^n$ is the coupling force due to the interaction with the other robot (e.g., the force feedback at the local side or the control force at the remote side) while $\boldsymbol{F}_w^{ext}(t)\in\mathbb{R}^n$ is the vector of generalized external forces, i.e., the force applied by the user, at the local side, or the reaction force applied by the environment, at the remote side. The kinetic energy of the system described in (5.47) is given by

$$K_w\left(\dot{\boldsymbol{x}}_w(t)\right)=\frac{1}{2}\dot{\boldsymbol{x}}_w(t)^T\boldsymbol{B}_w\left(\boldsymbol{x}_w(t)\right)\dot{\boldsymbol{x}}_w(t) \tag{5.48}$$

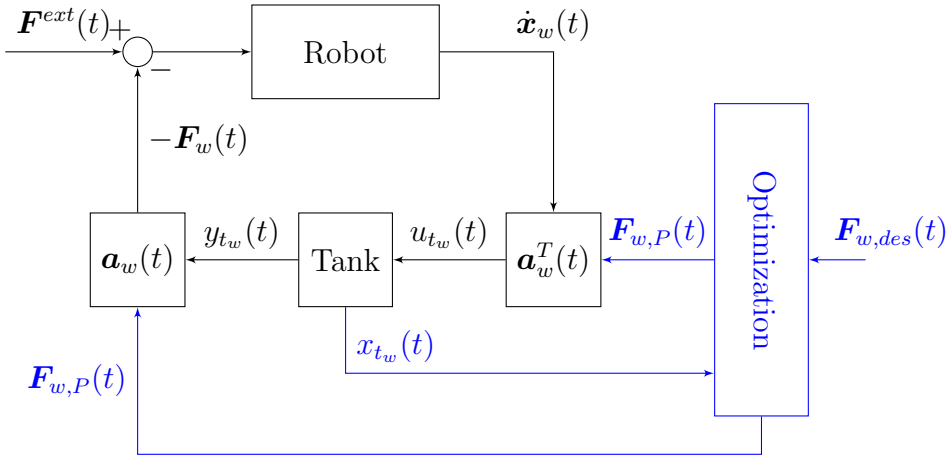


Figure 5.4: The control architecture for a robot

According to [125], from (5.47) and the properties of the Euler-Lagrange systems, the following balance holds

$$\dot{K}_w \leq \dot{\boldsymbol{x}}_w^T \Big(\boldsymbol{F}_w^{ext} + \boldsymbol{F}_w \Big) \tag{5.49}$$

which implies

$$K_w(t) - K_w(0) \leq \int_0^t \dot{\boldsymbol{x}}_w^T(\tau) \Big(\boldsymbol{F}_w^{ext}(\tau) + \boldsymbol{F}_w(\tau) \Big) d\tau \tag{5.50}$$

This means that (5.47) is passive with respect to the pair $(\boldsymbol{F}_w^{ext} + \boldsymbol{F}_w, \dot{\boldsymbol{x}}_w)$ with the kinetic energy $K(t)$ as storage function. Local and remote robots can exchange information through a communication channel characterized by a constant delay $\delta > 0$. We aim at building a bilateral teleoperation architecture that is passive independently of δ and that, given any desired value $\boldsymbol{F}_{w,des}(t)$ for $\boldsymbol{F}_w(t)$, implements its best passive command. The proposed architecture provides a robustly stable behaviour and optimal performance. This is due to the passivity of the overall controlled system and thanks to the Quadratic Programming (QP) modulation policy which will be discussed later.

5.3.1 Modulated tank based control

The passive energetic coupling between local and remote sides, shown in Figure 5.4, is performed using the following energy tank

$$\begin{cases} \dot{x}_{t_w} = u_{t_w} \\ y_{t_w} = \frac{\partial T_w}{\partial x_{t_w}} \end{cases} \tag{5.51}$$

and the modulated interconnection

$$\begin{cases} u_{t_w}(t) = \boldsymbol{a}_w^T(t) \dot{\boldsymbol{x}}_w(t) \\ -\boldsymbol{F}_w(t) = \boldsymbol{a}_w(t) y_{t_w}(t) = \boldsymbol{a}_w(t) x_{t_w}(t) \end{cases} \tag{5.52}$$

where $(\dot{\boldsymbol{x}}_w(t), y_w(t)) \in \mathbb{R}^n \times \mathbb{R}^n$ is the input/output port and $\boldsymbol{a}_w(t) \in \mathbb{R}^n$ is defined as

$$\boldsymbol{a}_w(t) = -\frac{\boldsymbol{\gamma}_w(t)}{x_{t_w}(t)} \tag{5.53}$$

where $\boldsymbol{\gamma}_w(t) \in \mathbb{R}^n$ is the desired value for the output $\boldsymbol{F}_w(t)$. The modulated interconnection (5.52) has a singularity when $x_{t_w} = 0$. This is due to the fact that, from (5.6), when $x_{t_w}(t) = 0$, there is no more energy available in the tank and, therefore, no action can be implemented. Thus, in order to avoid singularities in (5.53), it is necessary to initialize x_{t_w} in such a way that $T_w(x_{t_w}(0)) = \varepsilon > 0$ and to guarantee that $T_w(x_{t_w}(t)) \geq \varepsilon \ \forall t > 0$. Again, it is also important to correctly upper bound $T_w(x_{t_w}(t))$ for preventing the implementation of practically unstable behaviours. In the following, for keeping the notation simple, we do not explicitly consider the saturation due to such upper bound. In fact, as shown in the previous section, this can be done by using the switching variable σ that bounds the energy stored. All the results are still valid when considering the saturation.

Proposition 1. *If $T_w(t) \geq \varepsilon$, $\forall t \geq 0$, the system consisting of the interconnection of* (5.47) *and* (5.51) *through* (5.52) *is passive with respect to the pair $(\boldsymbol{F}_w^{ext}, \dot{\boldsymbol{x}}_w)$.*

Proof. From (5.51) and (5.6) we can define

$$\dot{T}_w(t) = u_{t_w}(t) y_{t_w}(t) \tag{5.54}$$

and substituting (5.52) in (5.54) we get

$$\dot{T}_w = \boldsymbol{a}_w^T(t)\dot{\boldsymbol{x}}_w(t) y_{t_w}(t) = \Big(\boldsymbol{a}_w(t) y_{t_w}(t)\Big)^T \dot{\boldsymbol{x}}_w(t) = -\boldsymbol{F}_w(t)^T \dot{\boldsymbol{x}}_w(t) \tag{5.55}$$

Consider the non negative storage function $Q_w(t)$ defined as

$$Q_w(t) = K_w(t) + T_w(t) \tag{5.56}$$

where $K_w(t)$ is the kinetic energy of the robot defined in (5.48). Using (5.49) and (5.55), we have that

$$\dot{Q}_w(t) = \dot{K}_w(t) + \dot{T}_w(t) \leq \dot{\boldsymbol{x}}_w^T\Big(\boldsymbol{F}_w^{ext} + \boldsymbol{F}_w\Big) - \dot{\boldsymbol{x}}_w^T \boldsymbol{F}_w = \dot{\boldsymbol{x}}_w^T \boldsymbol{F}_w^{ext} \tag{5.57}$$

which implies the following passivity condition

$$\int_0^t \dot{\boldsymbol{x}}_w^T(\tau) \boldsymbol{F}_w^{ext}(\tau) d\tau \geq Q_w(t) - Q_w(0) \geq -Q_w(0) \tag{5.58}$$

□

From (5.55) it follows that the energy necessary for implementing $\boldsymbol{F}_w$ is extracted/injected from/into the tank. If $\boldsymbol{F}_w\dot{\boldsymbol{x}}_w \leq 0$, then the desired control action is dissipative and the corresponding energy is stored in the tank. On the other hand, if $\boldsymbol{F}_w\dot{\boldsymbol{x}}_w > 0$, then the desired control action requires energy for being implemented and this energy is extracted from the tank. It is possible to harvest some energy for refilling the tank by adding a damping term, κ_w, to the desired value for the output. This can be done by setting $\boldsymbol{\gamma}_w = \Gamma - \kappa_w\dot{\boldsymbol{x}}_w$ in (5.53), where Γ is the desired value for the modulation, i.e., for $\boldsymbol{F}_w(t)$. This leads to the following balance

$$\dot{T}_w = -\dot{\boldsymbol{x}}_w^T\boldsymbol{F}_w + \dot{\boldsymbol{x}}_w^T\kappa_w\dot{\boldsymbol{x}}_w \tag{5.59}$$

where the second term of the right-hand side is always positive and, therefore, it brings energy into the tank by extracting it from the robot. Thus using the energy tank, as illustrated in Figure 5.4, allows to exploit the energy of the tank for reproducing the desired behaviour. As shown in Proposition 1, the system controlled by the tank is passive as long as $T_w(t) \geq \varepsilon$. Thus, the energy stored in the tank is that we have for passively implementing the desired control action. This means that leaving some energy in the tank becomes a constraint for finding the best passive implementation of the desired control input. In Figure 5.4, the full control architecture of the local/remote robot is the union of the robot/tank system and the optimization block, which is responsible for modulating the interconnection between the tank and the robot, i.e., to find the optimal value for $\boldsymbol{\gamma}_w$ in (5.53).

Given a desired value $\boldsymbol{F}_{w,des}$ for the control input of the robot, the optimization block, considering the amount of energy available in the tank $T_w(x_{t_w})$, finds the best passive implementation of the desired control input $\boldsymbol{F}_{w,P}$, by solving the optimization problem

$$\boldsymbol{F}_{w,P}(t) = \underset{\bar{\boldsymbol{F}}_w(t)}{\arg\min} \quad \|\bar{\boldsymbol{F}}_w(t) - \boldsymbol{F}_{w,des}(t)\|_2^2 \tag{5.60}$$

$$\text{subject to} \quad T_w(x_{t_w}(t)) \geq \varepsilon \tag{5.61}$$

and by setting $\boldsymbol{\gamma}_w = \boldsymbol{F}_{w,p}$. Exploiting the balance (5.55) and considering also that $\boldsymbol{F}_w(t) = \boldsymbol{F}_{w,P}(t)$ (5.60) can be rewritten as

$$\boldsymbol{F}_{w,P}(t) = \underset{\bar{\boldsymbol{F}}_w(t)}{\arg\min} \quad \|\bar{\boldsymbol{F}}_w(t) - \boldsymbol{F}_{w,des}(t)\|_2^2 \tag{5.62}$$

$$\text{subject to} \quad \int_0^t -\dot{\boldsymbol{x}}_w^T(\tau)\boldsymbol{F}_{w,P}(\tau)d\tau \geq -T_w(x_{t_w}(0)) + \varepsilon \tag{5.63}$$

As shown in [122], the problem (5.62) can be reformulated as a convex optimization problem in discrete-time and, as shown in Section 5.3.3, it is amenable to be solved in real-time.

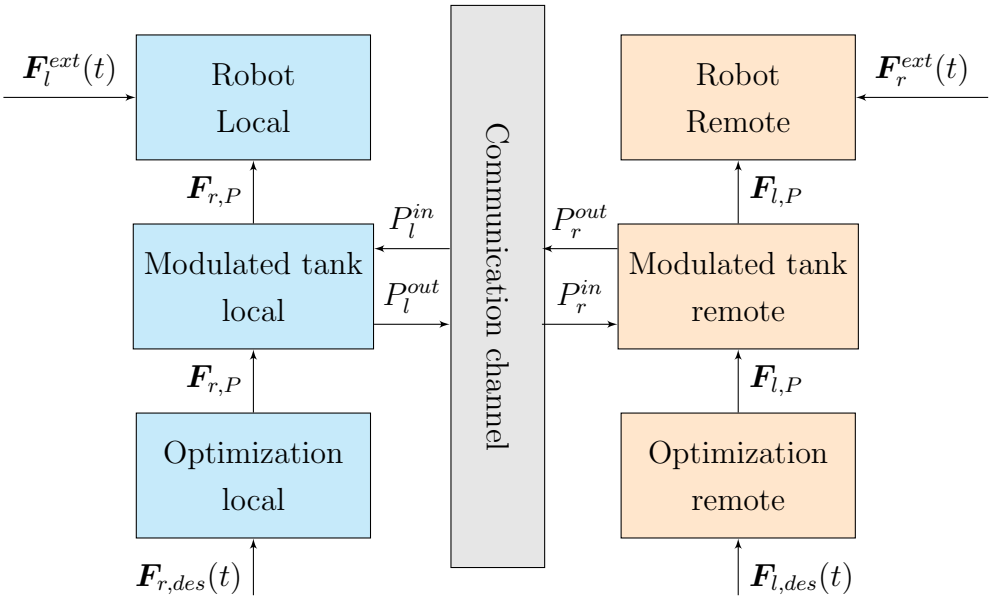


Figure 5.5: The optimized two-layer architecture for bilateral teleoperation

5.3.2 Optimised two-layer architecture

The whole control architecture is shown in Figure 5.5 and each side of the teleoperation architecture is modelled as

$$\begin{cases} \boldsymbol{B}_w\left(\boldsymbol{x}_w(t)\right)\ddot{\boldsymbol{x}}_w(t) + \boldsymbol{C}_w\left(\boldsymbol{x}_w(t), \dot{\boldsymbol{x}}_w(t)\right)\dot{\boldsymbol{x}}_w(t) + \boldsymbol{D}\dot{\boldsymbol{x}}_w(t) = -\boldsymbol{a}_w(t)x_{t_w}(t) + \boldsymbol{F}_w^{ext}(t) \\ \dot{\boldsymbol{x}}_{t_w} = \boldsymbol{a}_w^T(t)\dot{\boldsymbol{x}}_w(t) + \frac{1}{x_{t_w}}(P_w^{in} - P_w^{out}) \end{cases} \tag{5.64}$$

where $P_w^{in}, P_w^{out} \geq 0$ are the incoming and the outgoing power flows, as defined in the previous section. Considering the storage function in (5.56) and following the same procedure outlined in Proposition 1, it is possible to derive

$$\dot{Q}_w(t) \leq \dot{\boldsymbol{x}}_w^T \boldsymbol{F}_w^{ext} + P_w^{in} - P_w^{out} \tag{5.65}$$

and we can thus show that the teleoperation system represented in Figure 5.5 is passive independently of the communication delay.

Proposition 2. *Considering the teleoperation system where the local and remote sides are defined by* (5.64) *and they exchange power according to* (5.29)*. The overall teleoperation is passive with respect to the pair* $((\boldsymbol{F}_l^{ext}, \boldsymbol{F}_r^{ext}), (\dot{\boldsymbol{x}}_l, \dot{\boldsymbol{x}}_r))$.

Proof. The power flowing in the communication channel is given by

$$P_{ch}(t) = P_l^{out}(t) - P_l^{in}(t) + P_r^{out}(t) - P_r^{in}(t) \tag{5.66}$$

Considering (5.29), we can rewrite (5.66) as

$$P_{ch}(t) = P_l^{out}(t) - P_l^{out}(t-\delta + P_r^{out}(t) - P_r^{out}(t-\delta) =: \dot{H}_{ch} \tag{5.67}$$

and finally we get

$$\dot{H}_{ch}(t) = \frac{d}{dt}\int_{t-\delta}^{t}(P_l^{out}(\tau) + P_r^{out}(\tau))d\tau \tag{5.68}$$

where $H_{ch}(t)$ represents the amount of energy stored in the communication channel during the exchange of power between the tanks. Since $P_l^{out}(t), P_r^{out}(t) \geq 0$, H_{ch} is non-negative. Let consider as a storage function the total energy of the teleoperation system

$$W(t) = \sum_{w\in\{l,r\}} Q_w(t) + H_{ch}(t) \tag{5.69}$$

where Q_w is defined in (5.56). Using (5.65), it is possible to write

$$\dot{W}(t) \leq \dot{\boldsymbol{x}}_l^T \boldsymbol{F}_l^{ext} + P_l^{in}(t) - P_l^{out}(t) + \dot{\boldsymbol{x}}_r^T \boldsymbol{F}_r^{ext} + P_r^{in}(t) - P_r^{out}(t) + \dot{H}_{ch} \tag{5.70}$$

and substituting (5.66) and (5.67) into (5.70), we have

$$\dot{W}(t) \leq \dot{\boldsymbol{x}}_l^T \boldsymbol{F}_l^{ext} + \dot{\boldsymbol{x}}_r^T \boldsymbol{F}_r^{ext} \tag{5.71}$$

and

$$\int_0^t \dot{\boldsymbol{x}}_l^T(\tau)\boldsymbol{F}_l^{ext}(\tau) + \dot{\boldsymbol{x}}_r^T(\tau)\boldsymbol{F}_r^{ext}(\tau)d\tau \geq W(t) - W(0) \tag{5.72}$$

which proves the passivity of the overall system. □

The control architecture has been illustrated for the case in which both local and remote subsystems have an admittance causality (i.e., force in/velocity out). In bilateral teleoperation, it often happens that we aim at commanding the remote robot to track a velocity input v_{des} (i.e., the velocity of the local robot). This can be done by setting

$$\boldsymbol{F}_{r,des} = \mathcal{C}(v_{des}(t) - \dot{\boldsymbol{x}}_r) \tag{5.73}$$

where $\mathcal{C}$ represents a generic controller enabling the tracking of v_{des} (e.g., a PD controller). In case one of the robots is velocity controlled, all the presented theory is still valid. In fact, making the common assumption that $\dot{\boldsymbol{x}}_w \approx u_w$, where $u_w \in \mathbb{R}^n$ is the velocity input, we can possible model the velocity controlled robot as a simple integrator

$$\begin{cases} \dot{\boldsymbol{x}}_w = u_w \\ y_w = \boldsymbol{x} \end{cases} \tag{5.74}$$

that is passive with respect to the pair (u_w, y_w). It is possible to couple a tank to (5.74) and to optimally exploit the energy stored in the tank for implementing the best passive input.

5.3.3 Experimental validation

To evaluate the efficiency and the robustness of the proposed architecture, two experiments have been done. In the first experiment, a torque-controlled manipulator has been used as a remote robot, and a puncturing-like task has been implemented. In the second experiment, the dVRK is exploited for the peg-in-hole training task. This allows validating the effectiveness of the proposed architecture in case the remote robot is velocity controlled. For both experiments, a communication delay of 200 ms between the local and remote sides has been considered. The same parameters have been used for local and remote controllers ($T_w^{max} = 5.0\ J$, $T_w^{ava} = 3.0\ J$,$T_w^{init} = 2.5\ J$, $T_w^{req} = 2.0\ J$, $\bar{P} = 0.05\ W$, $\varepsilon = 0.01\ J$).

Experiment with a torque controlled manipulator

In this experiment, the user controls the remote robot in order to explore the environment populate both by soft and by rigid objects. The local robot is a 6-DOF OMEGA.6 haptic device and the remote robot is a KUKA LWR 4+ 7-DOF torque-controlled robot endowed with an internal Force/Torque sensor. The desired force feedback is the interaction force measured at the remote side, i.e., $\boldsymbol{F}_{l,des}(t) = \boldsymbol{F}_r^{ext}(t-\delta)$. The remote robot is velocity controlled to track $\boldsymbol{v}_{des}(t) = \dot{\boldsymbol{x}}_l(t-\delta)$. Figure 5.6a shows the Cartesian position of the local and the remote robots along the x-axis. It can be noticed that the position tracking is very accurate in free motion. During the interaction, there is a mismatch between the positions because the remote robot cannot follow the local one but once the interaction is over a good tracking is re-established. Figure 5.6b shows that the velocity of the remote robot tracks in a good way the local velocity, while satisfying the passivity constraint imposed in (5.61). In Figure 5.6c, the interaction force measured by the remote sensor and the force feedback implemented at the local side are shown. The action of the optimizer can be seen around 20 s, 30 s to 35 s and 35 s to 40 s when the energy within the local tank is not enough to implement the desired remote force. This modulation proves the effectiveness of the proposed architecture.

Figure 5.6d and Figure 5.7 shows the evolution of the energy, the energy requests and the outgoing power. The energy in the tanks changes according to the interaction between robot and environment and to the exchange of internal energy and power. It is possible to notice how the energy inside the local tank has an alternating evolution because when the force feedback needs to be applied, some energy of the tank has to be used. At the remote side the energy drops due to interactions with the environment and to execute the

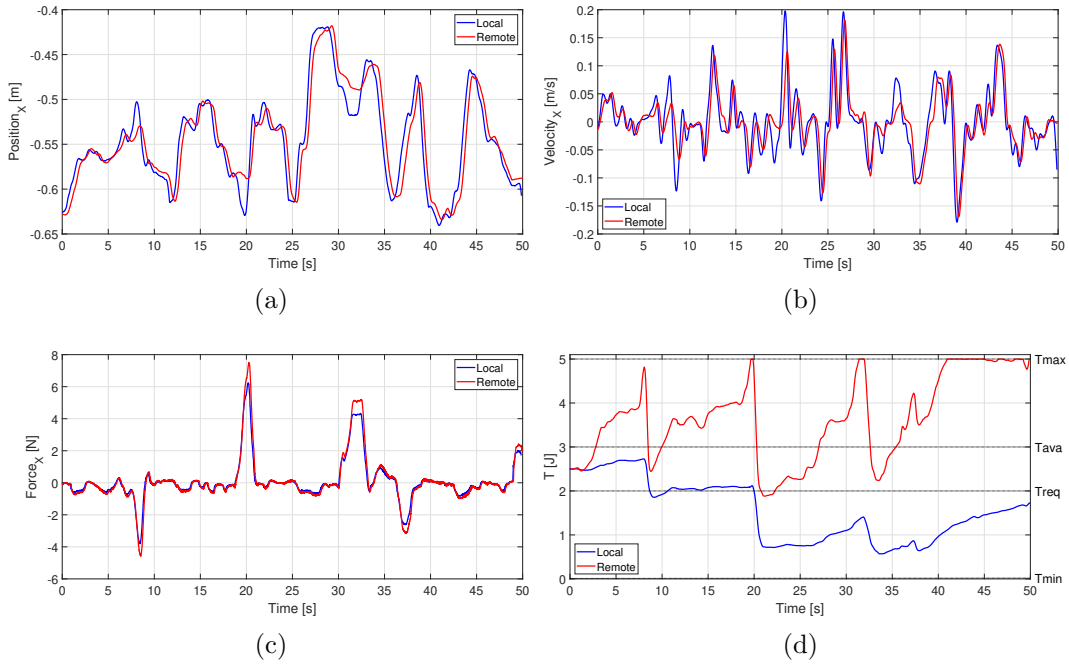


Figure 5.6: The experimental validation using a torque controlled manipulator (data are reported only along the x-axis). a,b) the Cartesian position and velocity of the local and remote manipulators, c) the local side force feedback and the remote side interaction force, d) the energy evolution of the local and remote side tank.

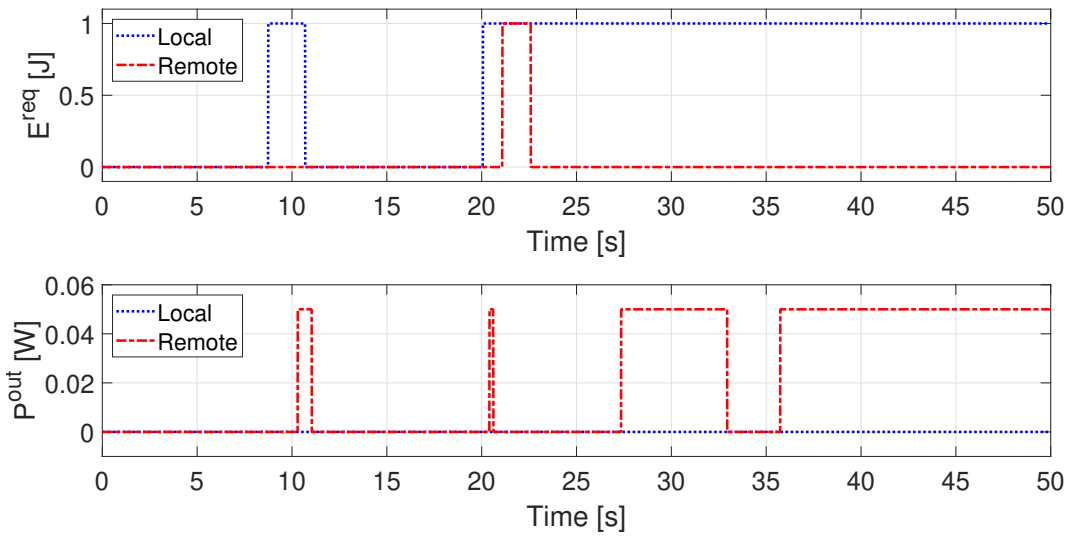


Figure 5.7: Energy requests and power flows between the energy tanks. In dotted blue line the local-to-remote energy flow and in dashed red line the vice versa.

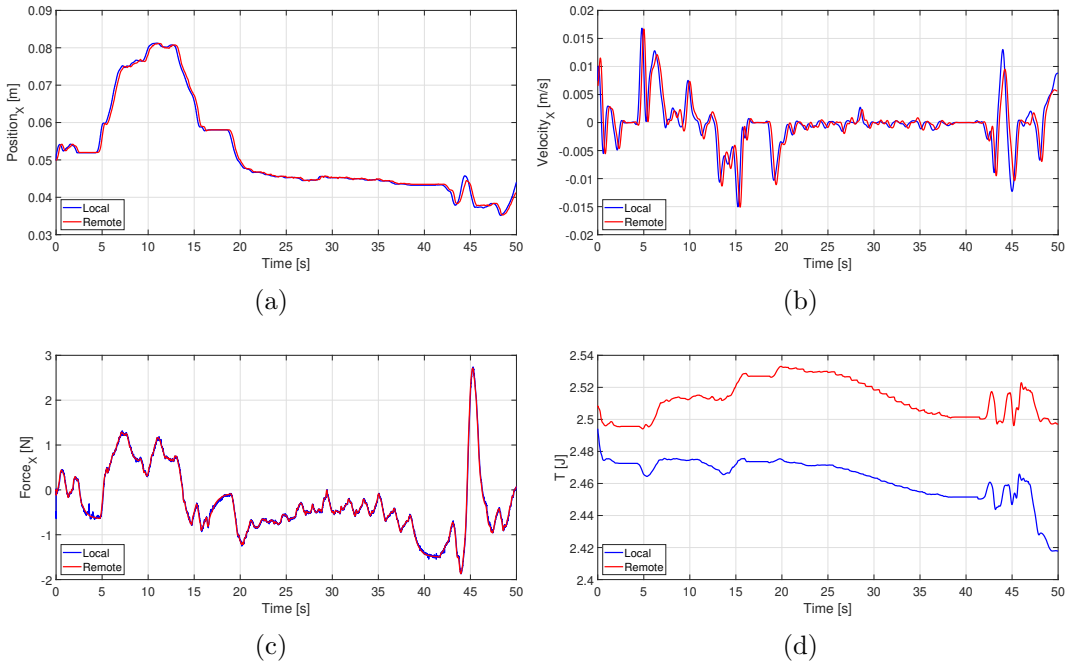


Figure 5.8: The experimental validation using a surgical robotic platform (data are reported only along the x-axis). a,b) the Cartesian position and velocity of the local and remote manipulators, c) the local force feedback and the remote interaction force, d) the energy evolution of the local and remote side tank.

control action necessary for generating good velocity tracking.

Furthermore, when the energy of the local tank falls below the minimum level T_w^{req}, an energy request (in a constant way from $t = 20\,\mathrm{s}$) is sent. The remote side sends back to the local tank the power flow $\bar{P}$ when the energy level of the remote tank is greater than T_w^{ava}.

Experiment with the dVRK

The experiment is the classical peg-in-hole task [126] with the dVRK setup, using one MTM and one PSM. Because the holes have different tolerances, we expect to measure different levels of force during the peg insertion. This test will pave the way to more extensive experiments aimed at solving the long-standing issues related to using force feedback in robotic surgery.

The MTM is force-controlled while the PSM is velocity-controlled. The interaction force $\boldsymbol{F}_r^{ext}$ between the PSM and the environment is estimated by the PSM internal sensing system and is sent to the local side to produce the desired force feedback, i.e., $\boldsymbol{F}_{l,des}(t) = \boldsymbol{F}_r^{ext}(t-\delta)$. Since the PSM is velocity-controlled, it is modelled as (5.74) and the local energetic architecture is adapted accordingly.

Figure 5.8a shows the Cartesian position of the local and the remote along the x-axis. It can be easily seen how the remote robot (PSM) can replicate the position of the local robot (MTM) in a very precise way since the desired speed at the remote side has been correctly tracked, as can be seen in Figure 5.8b. The red line, relating to the remote velocity tracks in a good way the local velocity, blue line. It should be noted that the sudden evolution of the velocity is due both to the vibration produced by small motions performed by the user and by the measurement noise. However, it is important to highlight the correct functioning of the system. The force feedback implemented at the local side and the measured interaction force between the remote and the environment are shown in Figure 5.8c. It can be noticed that the force tracking is very good and this verifies the effectiveness of the proposed control architecture.

Figure 5.8d shows the evolution of the energy level in the tanks. Due to the limited workspace and of the limited velocities and forces the robots are experiencing during the experiment, the dynamics of the energy of the tanks is quite slow. The energy in the tanks remains always greater than the T^{req} threshold and, therefore, no energy requests are sent and, consequently, no power is transmitted over the communication channel. For this reason, they have not been reported. Nevertheless, the tanks never get empty and this is sufficient for guaranteeing a passive behaviour of the system and the possibility of implementing the desired inputs at local and remote sides.

Chapter 6

Optimal control for safe bilateral teleoperation

In Chapter 5 we exploited the *two-layers* bilateral teleoperation architecture to guarantee the take-over by the surgeon in case of unexpected events. Such control methodology, as we previously discussed, has at its core the decoupling between the control policy (the *transparency layer*) and the passivity policy (the *passivity layer*). However, such architecture can prevent a global optimal control strategy and cannot enforce constraints during the teleoperation. In fact, since the passivity action is made after the computation of the control action the performance in terms of transparency can be arbitrarily degraded to keep the system passive. For this reason, in this chapter we will investigate on the application of optimal control to the bilateral teleoperation.

First of all, we will define a non-linear MPC problem where the passivity is enforced as an optimization constraint together with the constraint of the maximum interaction forces at the remote side. This methodology has an important applications in the field of R-MIS. In fact, even if force feedback allows a surgeon to feel the remote interaction with the anatomical structures,

---

Part of this chapter is based on the following publications:

- N. Piccinelli and R. Muradore, "A passivity-based bilateral teleoperation architecture using distributed nonlinear model predictive control," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2020, pp. 11 466–11 472
- N. Piccinelli and R. Muradore, "A bilateral teleoperation with interaction force constraint in unknown environment using non linear model predictive control," *European Journal of Control*, vol. 62, pp. 185–191, 2021
- N. Piccinelli and R. Muradore, "Interaction force constraints for position-based manipulator using linear mpc," Under review
- N. Piccinelli and R. Muradore, "Control of interaction force using hybrid linear model predictive control," in *2022 Mediterranean Conference on Control and Automation (MED)*, 2022
- N. Piccinelli and R. Muradore, "Passivity-based bilateral teleoperation using linear model predictive force control," Under review

the surgeon cannot be fast enough to prevent unsafe interactions, in particular when communication delay affects the system. Unfortunately, the non-linear MPC cannot be employed for real-time control of manipulators due to the high computational effort required. For this reason, we will later present an approximated linear MPC solution which guarantees the passivity and meanwhile allows the real-time control of a robotic manipulator. It is worth mentioning that, the approximated dynamics due to the linearisation does not allow to vary the contact state at the remote side robot within the prediction horizon. To overcome this shortcoming, we investigated the modelling of the bilateral teleoperation as a hybrid system and we derived an MPC controller able to handle the switching behaviour of the interaction forces. Such methodology can be seen as a trade-off between model accuracy and computational burden.

In Section 6.1 we will review the state of the art of optimal control in bilateral teleoperation systems, while in Section 6.2 we will recall the model of the manipulator and of the environment. They will be used to define the MPC problems.

The non-linear MPC controller and its use within the bilateral teleoperation architectures will be presented in Section 6.3. We will validate the proposed control scheme in both simulated (multi-DOF manipulator) and real environment (1-DOf manipulator). We will also estimate the interaction forces using the generalized momentum observer that could be exploited in case a force sensor at the manipulator's end-effector is not available.

Sections 6.4 and 6.5 will design linear MPC controllers for multi-DOF manipulators. In particular, Section 6.4 will investigate on the feasibility of adopting linear MPC to perform direct force control for industrial manipulation, while Section 6.5 will focus on the definition of an under-approximated linearised tank dynamics. Finally we integrate the previous tools to design a linear passive MPC controller for bilateral teleoperation.

Finally in Section 6.6 Hybrid Systems (HS) will be used to properly model the interaction between the remote side manipulator and the environment. The proposed hybrid linear MPC will be tested in a simulated environment using a 1-DOF manipulator.

The validation of the methodologies proposed in this chapter has been performed on platforms different from the autonomous arms used in *SARAS*. Nonetheless, we plan to test such techniques with the Franka Emika arms soon.

6.1 State of the art

Optimal control and in particular MPC, has been already adopted in the past to design bilateral teleoperation architecture. For instance, in [132] input/output constraints are enforced using MPC with unbounded communication delay. In case of long delays the system switches in an open-loop mode; the MPC is only placed at the operator side. A different approach, presented in [133], incorporates the delay in the discrete state-space model of a Linear Quadratic Gaussian (LQG) controller to enhance transparency in case of known constant delay. The main drawback is that the state-space dimension is proportional to the delay. In [134] a model predictive controller is also used to achieve stable teleoperation with constant but unknown time delays and force feedback. The delay effect is managed by designing specific controllers for the free-motion and contact scenarios. In [135] the authors adopts a single MPC at the operator side allowing teleoperation under time-varying delay when the state at the environment robot is estimated using a Smith predictor.

In [136] a switched Cartesian force control teleoperation was used to perform a tele-echography. The method uses a 3D time-of-flight camera to detect the contact and a force sensor to estimate the environment model parameters. In [137] an integration of contact force and virtual spring is proposed. This method allows performing teleoperation in case of constant-time delay choosing the appropriate stability margin. The usage of the MPC in recent years has been explored as a way to perform safe interaction integrating impedance [138] and admittance control [139]. Finally in [140] an MPC has been used for position/force trajectory tracking during the interaction of the remote manipulator with the environment.

Manipulation usually requires multiple interactions between the manipulator and the surrounding environment, causing the dynamics of the system to switch between free motion and contact states. The resulting dynamics can decrease the stability margin or generate chattering due to the large impulsive forces that may occur during the transition phase between free and constrained motion with, e.g. hard environment. To cope with these behaviours many control schemes have been proposed. For instance, impedance control defines a dynamic relationship between the position of the end-effector and the force exerted on the environment [141]. Other approaches exploit switched control laws where parallel position/force control is used in contact and position control in free-motion [142]. In [143] a combination of a compliant wrist design with a switching position/force approach for time-varying trajectory tracking has been proposed. Such a solution, suitable in scenarios like bilateral teleop-

eration, provides stability of the transition phase and accurate force tracking. In [144] a force regulation methodology reduces the number of bounces by means of a discontinuous transition control algorithm. Another solution to improve the transient behaviour and to reduce the force overshoot uses a non-linear damping controller [145].

To combine the continuous and discrete behaviours of the interaction force, the theory of hybrid systems is often exploited. In [146], using hybrid stability theory, the asymptotic stability of an 1-DOF manipulator controlled with a switched PD-PI controller has been proved. Impedance controller for discrete-continuous dynamics modelling of multi-fingered manipulation is implemented in [147]. In [148], [149], hybrid controller with a hysteresis switching mechanism has been used to improve contact detection in a switched parallel position/force and pure position controller. A different approach to hybrid systems is implemented in [150], where stable contacts are obtained by making the contact phase a controlled invariant using constraints on the interaction force. Alternatively, Dynamical Systems (DS) modulate the robot's motion in such a way that stable contacts can be achieved [151]. In [152], optimal control has been used to find the optimal velocity to reduce the impact force. However, this could lead to very slow velocity with a reduction in transparency. Finally, in [153] a hybrid system has been used to model an unmanned aerial vehicle during interaction with walls. The system is then controlled through MPC achieving stable docking manoeuvres while minimizing rebounds.

Most of the robots nowadays are position-controlled since the human-robot interaction and/or the manipulation of fragile objects are not of primary importance. Recently, the so called collaborative robots are changing the manufacturing industry since many companies are now providing lightweight robots with complaint behaviour [154], [155]. However, the introduction of robots in challenging areas like surgery, rescue or space exploration requires the development of more sophisticated control techniques. For instance, in [156] a DS based technique adapts the motion in case of contact tasks to be robust against external disturbances. The main drawbacks of such methodology is the non-penetrable environment assumption and the lack of precise force tracking. In [157] the good performance while executing cooperative tasks with humans are obtained by a variable impedance control. In [158] a neural network admittance-based control has been designed for the generation and tracking of reference trajectories in a constrained task space.

The adoption of MPC based solution for real-time control is challenging due to the computational effort required to solve the optimization problem at each control cycle. In [138] an impedance controller has been designed in

terms of an MPC control problem. This solution provides compliant behaviour while interacting with the environment by enforcing constraints on the robot's speed, energy or jerk. In the case of position-based manipulators, the same compliant behaviour can be achieved using an MPC-based admittance controller [139]. Such methodology prevents the instability issues due to the high speed operation in case of high contact force magnitudes. Admittance control has been also exploited in [159], [160] to perform path following with a desired admittance or with direct force control. Moreover, in [161] such technique has been extended to mobile robots to enforce an interaction force limit.

As we showed before, the interaction forces are explicitly modelled within a non-linear MPC in the context of bilateral teleoperation with the aim of constraining them to provide safe interaction with the environment. The same modelling idea was proposed in [162] where a robotic manipulator is controlled by an MPC for tracking both position and force references. Recently, the idea of the modelling the environment within the state variable has been used in [140]. In this case the MPC guarantees the position/force tracking during the interaction of the manipulator with the environment. Finally, in [163] the variability and uncertainty of the environment are modelled as Gaussian processes and an observer is designed to estimate the contact forces applied by an MPC.

6.2 Robot model and environment estimation

Before proceedings with the definition of the optimal bilateral teleoperation, we will introduce the linearized manipulator' dynamics and the methodology used to estimate the model of the environment.

6.2.1 Joint space feedback linearization

Let the dynamic model of a gravity compensated robot be

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{q}} = \boldsymbol{u} + \boldsymbol{\tau} \tag{6.1}$$

where $\boldsymbol{q} \in \mathbb{R}^n$ represents the joint space coordinates, $\boldsymbol{u}$ is the command input, $\boldsymbol{\tau}$ is the generalized external torque, $\boldsymbol{M}(\boldsymbol{q})$ is the inertia matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is the centrifugal and Coriolis terms and $\boldsymbol{B}(\boldsymbol{q}) \succcurlyeq 0$ is a positive semi-definite matrix corresponding to friction and, if needed, artificial damping terms [124]. Moreover, the matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ satisfies the following passivity property

$$\dot{\boldsymbol{M}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})^T. \tag{6.2}$$

The generalized external torque $\boldsymbol{\tau}$ corresponds to the Cartesian interaction wrench with the environment or with the operator $\boldsymbol{h}_{ext}$

$$\boldsymbol{\tau} = \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{h}_{ext} \tag{6.3}$$

where $\boldsymbol{J}(\boldsymbol{q})$ is the Jacobian matrix of the manipulator. The robot model we will use within the MPC is the exact linearisation via feedback of the robot dynamics (6.1) [164]. Then each degree of freedom will be independent of the others and modelled as a mass-damper system

$$j_i\ddot{q}_i + b_i\dot{q}_i = u_i + \tau_i, \quad \forall i = 1 \dots n \tag{6.4}$$

where $b_i > 0$ is the friction and $j_i > 0$ is the inertia. The equivalent system in the configuration space is

$$\boldsymbol{J}\ddot{\boldsymbol{q}} + \boldsymbol{B}\dot{\boldsymbol{q}} = \boldsymbol{u} + \boldsymbol{\tau} \tag{6.5}$$

where $\boldsymbol{q} = \begin{bmatrix} q_1 \dots q_n \end{bmatrix}^T$, $\dot{\boldsymbol{q}} = \begin{bmatrix} \dot{q}_1 \dots \dot{q}_n \end{bmatrix}^T$, $\boldsymbol{u}$ and $\boldsymbol{\tau}$ are the control and external torques, $\boldsymbol{J} = \text{diag}(j_1 \dots j_n)$ and $\boldsymbol{B} = \text{diag}(b_1 \dots b_n)$.

Let $h_i(q_i, \dot{q}_i, u_i)$ be the i-th update function of the equivalent discrete-time second-order linear system

$$\begin{bmatrix} q_i(k+1) \\ \dot{q}_i(k+1) \end{bmatrix} = h_i\left(\begin{bmatrix} q_i(k) \\ \dot{q}_i(k) \end{bmatrix}, u_i(k), \tau_i(k)\right) = \bar{\boldsymbol{A}}_i \begin{bmatrix} q_i(k) \\ \dot{q}_i(k) \end{bmatrix} + \bar{\boldsymbol{B}}_i(u_i(k) + \tau_i(k)) \tag{6.6}$$

where q_i is the joint position, $\dot{q}_i$ is the joint velocity, u_i is the torque applied by the motor, τ_i is the external force and k denotes $t_k = k\Delta t$ with Δt is the sample time. This implies that the state space matrices $\bar{\boldsymbol{A}}_i$ and $\bar{\boldsymbol{B}}_i$ are the discretised version of the SISO differential equation (6.4). We indicate with $h(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u})$ the following multi DOF update function

$$\begin{bmatrix} \boldsymbol{q}(k+1) \\ \dot{\boldsymbol{q}}(k+1) \end{bmatrix} = h\left(\begin{bmatrix} \boldsymbol{q}(k) \\ \dot{\boldsymbol{q}}(k) \end{bmatrix}, \boldsymbol{u}(k), \boldsymbol{\tau}(k)\right) = \bar{\boldsymbol{A}} \begin{bmatrix} \boldsymbol{q}(k) \\ \dot{\boldsymbol{q}}(k) \end{bmatrix} + \bar{\boldsymbol{B}}(\boldsymbol{u}(k) + \boldsymbol{\tau}(k)). \tag{6.7}$$

The external force at the operator side $\boldsymbol{\tau}_o$ is assumed to be a measurable input of the system. The interaction $\boldsymbol{\tau}_r$ between the remote side robot and the environment is modelled as an unknown dynamic system.

6.2.2 Cartesian space feedback linearization

The same linearization methodology can be implemented in the operative space, where the gravity compensated manipulator is modelled by the following Euler-Lagrangian system

$$\boldsymbol{M}(\boldsymbol{x})\ddot{\boldsymbol{x}} + \boldsymbol{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})\dot{\boldsymbol{x}} + \boldsymbol{D}(\boldsymbol{x})\dot{\boldsymbol{x}} = \boldsymbol{u} + \boldsymbol{h}_{ext} \tag{6.8}$$

where $\boldsymbol{x} \in \mathbb{R}^6$ represents the Cartesian position and orientation, $\dot{\boldsymbol{x}}$ represents the Cartesian linear and angular velocities, $\boldsymbol{u}$ and $\boldsymbol{h}_{ext}$ are the control input and the external torque, respectively. $\boldsymbol{M}(\boldsymbol{x})$ is the inertia matrix, $\boldsymbol{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ is the centrifugal and Coriolis terms and $\boldsymbol{D}(\boldsymbol{x}) \succcurlyeq 0$ is the matrix containing friction terms and the artificial damping. As before the MPC will use a linearized model of the manipulator obtained by the exact linearisation via feedback of the robot dynamics. In this case each Cartesian coordinate is independent of the other and modelled as a mass-damper equation. To obtained such requirements we run our controller on top of an inner control loop that makes the robot dynamics linear exploiting the inverse dynamic technique [164]. Let the decoupled linearized Cartesian robot model be

$$\boldsymbol{M}\ddot{\boldsymbol{x}}(t) + \boldsymbol{D}\dot{\boldsymbol{x}}(t) = \boldsymbol{u}(t) + \boldsymbol{h}_{ext}(t) \tag{6.9}$$

such that the i-th coordinate is modelled as

$$m^i\ddot{x}^i + d^i\dot{x}^i = u^i + \tau^i \tag{6.10}$$

where $\boldsymbol{x}, \dot{\boldsymbol{x}} \in \mathbb{R}^n$, with $n = 6$, are the Cartesian position and velocity respectively, $\boldsymbol{u}$ and $\boldsymbol{h}_{ext}$ are the control and the external torques. $\boldsymbol{M} = \mathrm{diag}(j^1 \dots j^n)$ and $\boldsymbol{D} = \mathrm{diag}(b^1 \dots b^n)$ are the desired dynamic parameters along each coordinate. As before, the external force $\boldsymbol{h}_{ext} = \boldsymbol{h}_o$ is at the operator side is measured, while $\boldsymbol{h}_{ext} = \boldsymbol{h}_r$ is derived from the model of the interaction between manipulator and environment.

6.2.3 Environment model

The linear contact model of Kelvin-Voigt [165] is the most simple material model used for a rubber-like environment. This kind of material shows pure elastic behaviour in case of slow deformation and additional linear resistance to fast deformation. Under these assumptions, an environment modelled as a Kelvin-Voigt viscoelastic material can be represented as a pure viscous damper and a purely elastic spring connected in parallel.

For this reason, the interaction force $\boldsymbol{h}_r$ between the remote robot and the environment can be estimated using an identified first-order dynamic system having the position of the remote side manipulator $\boldsymbol{x}_r$ as input. The position of the environment $\boldsymbol{x}_e$ is estimated online as the position of the end-effector at the time of the first collision with the environment

$$\boldsymbol{x}_e = \boldsymbol{x}_r(k), \quad \text{if } \psi(k) = 1 \text{ and } \psi(k-1) = 0 \tag{6.11}$$

where $\boldsymbol{x}_r$ is the remote robot position and $\psi(k)$ is a Boolean function defined as follows

$$\psi(k) = \begin{cases} 1, & \text{if } |\boldsymbol{h}_e(k)| > T_e \\ 0, & \text{otherwise} \end{cases} \tag{6.12}$$

where T_e is a threshold that takes into account the measurement noise on $\boldsymbol{h}_e$. The equation of motion of the environment is modelled as

$$\boldsymbol{h}_r(t) = \boldsymbol{K}_e(\boldsymbol{x}_e - \boldsymbol{x}_r(t)) - \boldsymbol{B}_e\dot{\boldsymbol{x}}_r(t) \tag{6.13}$$

where $\boldsymbol{K}_e$ and $\boldsymbol{B}_e$ are positive definite matrices containing the unknown stiffness and damping parameters, respectively. These parameters are estimated online using the recursive least square method (RLS) when the robot at the remote side is in contact with the environment, and the parameters are kept constant along the prediction horizon in the next MPC control cycle.

6.3 Passivity-based non-linear MPC

In the following we will define the passivity-based MPC (P-MPC) for a robotic manipulator within a bilateral teleoperation architecture where an upper bound on the interaction force is considered. We will show its implementation first in a simple case (1-DOF) both in simulation and real environment (Sections 6.3.4 and 6.3.5) and then in a simulated multi-DOF scenario (Section 6.3.6). The interaction forces are estimated exploiting the generalised momentum approach proposed in [166], [167]. The overall control architecture is sketched in Figure 6.1, and the plant model within the P-MPCs (operator/environment) consists of the robot dynamics, the environment and the tank evolution. The control technique replicates at both sides of the teleoperation the whole system. We rewrite the tank dynamic defined in (5.27)

$$\dot{x}_{t_w} = \frac{\dot{\boldsymbol{x}}_w^T\boldsymbol{D}_w\dot{\boldsymbol{x}}_w + \sigma_w P_w^{in}(t)}{x_{t_w}} - \frac{P_w^{out}(t)}{x_{t_w}} + u_{t_w} \tag{6.14}$$

as the update function $g(E, \dot{\boldsymbol{q}}, \boldsymbol{u})$ discrete version of a non linear discrete-time system

$$\begin{aligned} E(k+1) &= g(E, \dot{\boldsymbol{q}}, \boldsymbol{u}) \\ &= E(k) + \Delta t((-\frac{1}{2}\dot{\boldsymbol{q}}^T\boldsymbol{B}\dot{\boldsymbol{q}} + \sigma P^{in}(k)) - P^{out}(k) + P^u(k)) \end{aligned} \tag{6.15}$$

where $E(k) \in \mathbb{R}^+$ represents the energy stored in the tank at time $t = k\Delta t$, $P^{in}(k)$, $P^{out}(k)$ are the previously defined power functions, and $P^u(k)$ is the power needed to apply the control input $\boldsymbol{u}$ during the optimisation

$$P^u(k) = -\boldsymbol{u}^T\dot{\boldsymbol{q}} \tag{6.16}$$

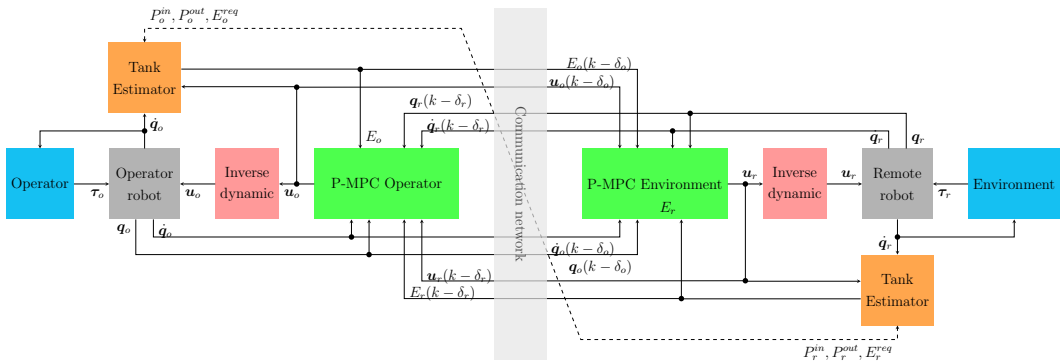


Figure 6.1: The bilateral teleoperation control architecture. The green blocks are the P-MPC controllers, the blue blocks are the sources of external torques, the grey blocks are the robots, the orange blocks are the tank estimators and the dashed line is the communication channel where P^{out} and E^{req} are exchanged.

The damping term $\boldsymbol{D}(\boldsymbol{q})$ in (6.1) in our case corresponds to the friction term and is equal to $\frac{1}{2}\dot{\boldsymbol{q}}^T\boldsymbol{B}\dot{\boldsymbol{q}}$.

6.3.1 Environment torque estimation

In case the manipulator does not have a force/torque sensor, as in the case of da Vinci® robot, the environment torque $\boldsymbol{\tau}_r$ can be estimated by means of the generalised momentum observer [167]. Starting from the linearised dynamic model (6.5) we define the generalised momentum of the overall manipulator as

$$\boldsymbol{p} = \boldsymbol{J}\dot{\boldsymbol{q}}. \tag{6.17}$$

The time derivative of the generalised momentum can be obtained by (6.5) and the passivity condition (6.2)

$$\dot{\boldsymbol{p}} = \boldsymbol{u} + \boldsymbol{\tau}_r - \boldsymbol{B}\dot{\boldsymbol{q}}. \tag{6.18}$$

The first-order momentum observer is derived from the residual vector [168]

$$\boldsymbol{r}(t) = \boldsymbol{K}\left(\boldsymbol{p}(t) - \int_0^t (\boldsymbol{u}(s) - \boldsymbol{B}\dot{\boldsymbol{q}} + \boldsymbol{r}(s))ds - \boldsymbol{p}_0\right) \tag{6.19}$$

where $\boldsymbol{K}$ is a positive definite constant matrix and $\boldsymbol{p}_0$ is the initial value of the generalised momentum. Using (6.18), the time derivative of the residual

$$\dot{\boldsymbol{r}} = -\boldsymbol{K}(\boldsymbol{r} - \boldsymbol{\tau}_r) \tag{6.20}$$

provides a first-order filtered version of $\boldsymbol{\tau}_r$. This means that we can use $\boldsymbol{r}$ as the estimate $\hat{\boldsymbol{\tau}}_r$ of the external forces in the joint space. As defined in Section 6.2.3, the equation of motion of the environment is modelled as

$$\boldsymbol{h}_r(k) = \boldsymbol{J}(\boldsymbol{q})^{-T}\hat{\boldsymbol{\tau}}_r = w_r(\boldsymbol{x}_r, \dot{\boldsymbol{x}}_r) = \boldsymbol{K}_e(\boldsymbol{x}_e - \boldsymbol{x}_r(k)) - \boldsymbol{B}_e\dot{\boldsymbol{x}}_r(k) \tag{6.21}$$

where k denotes $t_k = k\Delta t$.

6.3.2 Operator side teleoperation

Let $\boldsymbol{z}_o \in \mathbb{R}^N$ be the state vector of the MPC at the operator side, where $N = (2n+1) + (2m+1) + 6$ with n, m the number of degrees of freedom of the operator and remote side robots. The full state is equal to

$$\boldsymbol{z}_o(k) = \begin{bmatrix} \boldsymbol{q}_o(k) \\ \dot{\boldsymbol{q}}_o(k) \\ E_o(k) \\ \boldsymbol{q}_r(k-\delta_o(k)) \\ \dot{\boldsymbol{q}}_r(k-\delta_o(k)) \\ E_r(k-\delta_o(k)) \\ \hat{\boldsymbol{\tau}}_r(k-\delta_o(k)) \end{bmatrix} \tag{6.22}$$

where $\boldsymbol{q}_o, \dot{\boldsymbol{q}}_o \in \mathbb{R}^n$, $\boldsymbol{q}_r, \dot{\boldsymbol{q}}_r \in \mathbb{R}^m$ are the generalised position and velocity, $E_o, E_r \in \mathbb{R}$ are the tank levels, $\hat{\boldsymbol{\tau}}_r \in \mathbb{R}^m$ is the generalised estimated environment torque, $\boldsymbol{u}_o \in \mathbb{R}^n$ is the operator robot control command and $\delta_o(k) \in \mathbb{N}$ is the communication delay from environment to operator side at time $t = k\Delta t$. To compute $\hat{\boldsymbol{\tau}}_r$ we also need the position $\boldsymbol{x}_r$ and velocity $\dot{\boldsymbol{x}}_r$ of the remote robot in the operational space. These information can be retrieved using the forward kinematics of the manipulators. The update function is

$$\boldsymbol{z}_o(k+1) = f_o(\boldsymbol{z}_o, \boldsymbol{u}_o) = \begin{bmatrix} h_o(\boldsymbol{q}_o(k), \dot{\boldsymbol{q}}_o(k), \boldsymbol{u}_o(k)) \\ g_o(E_o(k), \dot{\boldsymbol{q}}_o(k), \boldsymbol{u}_o(k)) \\ h_r(\boldsymbol{q}_r(k-\delta_o(k)), \dot{\boldsymbol{q}}_r(k-\delta_o(k)), \boldsymbol{u}_r(k-\delta_o(k))) \\ g_r(E_r(k-\delta_o(k)), \dot{\boldsymbol{q}}_r(k-\delta_o(k)), \boldsymbol{u}_r(k-\delta_o(k))) \\ \boldsymbol{J}_r^T w_r(\boldsymbol{x}_r(k-\delta_o(k)), \dot{\boldsymbol{x}}_r(k-\delta_o(k))) \end{bmatrix} \tag{6.23}$$

where h_o is the manipulator dynamics (6.7), g_o is the tank dynamics defined in (6.15) and w_r is the environment model. Due to the distributed approach, the states of the dynamic system at the opposite side and the last applied command (i.e., the remote side for the operator side) are received after $\delta_o(k)$ steps of delay.

Assuming the controlled robot fully observable, at time k we can directly recover the state at each control cycle and thus compute its tank level $E_o(k)$ using the observed state $\boldsymbol{q}_o(k)$, $\dot{\boldsymbol{q}}_o(k)$ and the control input $\boldsymbol{u}_o(k)$. At the same control cycle, due to the communication delay, the remote side tank level $E_r(k-\delta_o(k))$ and the environment torque estimation $\hat{\boldsymbol{\tau}}_r$ are updated using the delayed remote side robot state $\boldsymbol{q}_r(k-\delta_o(k))$, $\dot{\boldsymbol{q}}_r(k-\delta_o(k))$ and the delayed control input $\boldsymbol{u}_r(k-\delta_o(k))$. The update of $E_r(k-\delta_o(k))$ and $\hat{\boldsymbol{\tau}}_r$ are computed applying (6.15) and (6.21), respectively.

The cost function to be minimised will balance the position and velocity tracking errors between the operator and remote side and the control signal too. The tracking error is expressed in the operational space in order to cope with different kinematics of the haptic device and the remote robot. The cost function moderates the control input $\hat{\boldsymbol{u}}_o$ and its rate of change $\Delta\hat{\boldsymbol{u}}_o$. The control input is also transformed in the operational space in order to deal with the different measurement unit of the translational and rotational control actions. The resulting cost function at time k at the operator side is

$$\begin{aligned}\mathcal{C}_o(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}_o) = &\sum_{i=1}^{k_p} \hat{\boldsymbol{e}}(k+i)^T \boldsymbol{\Lambda}_p \hat{\boldsymbol{e}}(k+i) + \hat{\dot{\boldsymbol{e}}}(k+i)^T \boldsymbol{\Lambda}_v \hat{\dot{\boldsymbol{e}}}(k+i) + \\ &+ \sum_{j=1}^{k_c} \hat{\boldsymbol{u}}_o(k+j)^T \boldsymbol{\Lambda}_c \hat{\boldsymbol{u}}_o(k+j) + \Delta\hat{\boldsymbol{u}}_o(k+j)^T \boldsymbol{\Lambda}_d \Delta\hat{\boldsymbol{u}}_o(k+j) \quad (6.24)\end{aligned}$$

where

$$\hat{\boldsymbol{e}}(k+i) \triangleq \hat{\boldsymbol{x}}_o(k+i) - \hat{\boldsymbol{x}}_r(k - \delta_o(k) + i) \quad (6.25)$$

and

$$\hat{\dot{\boldsymbol{e}}}(k+i) \triangleq \hat{\dot{\boldsymbol{x}}}_o(k+i) - \hat{\dot{\boldsymbol{x}}}_r(k - \delta_o(k) + i). \quad (6.26)$$

The estimated Cartesian position of the operator $\hat{\boldsymbol{x}}_o$ and the remote robot $\hat{\boldsymbol{x}}_r$ are obtained thought the forward kinematics of the manipulators. The estimated Cartesian velocity of the operator and remote robots are equal to $\hat{\dot{\boldsymbol{x}}}_o(k) = \boldsymbol{J}_o(\boldsymbol{q}_o(k))\hat{\dot{\boldsymbol{q}}}_o(k)$ and $\hat{\dot{\boldsymbol{x}}}_r(k) = \boldsymbol{J}_r(\boldsymbol{q}_r(k))\hat{\dot{\boldsymbol{q}}}_r(k)$. The cost function is parameterised over $\boldsymbol{\Lambda}_p$, $\boldsymbol{\Lambda}_v$, $\boldsymbol{\Lambda}_c$ and $\boldsymbol{\Lambda}_d$ which are application-dependant semi-positive definite weight matrices, and over k_p and k_c which are the prediction and control horizons. We decide to have different length of horizons, $k_p > k_c$, because in case of $k_c \ll k_p$ the control problem has less optimisation variables and this yields to faster computations. The optimal control inputs $\hat{\boldsymbol{u}}_o^\star(k), \ldots, \hat{\boldsymbol{u}}_o^\star(k+k_c)$ are obtained as the solution of the finite-horizon optimal control problem

$$\begin{aligned}\hat{\boldsymbol{u}}_o^\star(k+i)|_{i=0}^{k_c} = \quad & \underset{\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}_o}{\arg\min} \quad \mathcal{C}_o(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}_o) \\ & \text{subject to} \quad \hat{\boldsymbol{z}}(k) = \boldsymbol{z}(k) \\ & \qquad\qquad\quad \hat{\boldsymbol{z}}(k+1) = f_o(\hat{\boldsymbol{z}}, \boldsymbol{u}_o) \\ & \qquad\qquad\quad \hat{E}_o \geq \epsilon_o \quad (6.27) \\ & \qquad\qquad\quad \hat{E}_r \geq \epsilon_r \quad (6.28)\end{aligned}$$

where we assume $\boldsymbol{u}_r$ and $\boldsymbol{\tau}_o$ constants over the optimisation horizon and $\boldsymbol{z}(k)$ is the known state vector at time k. The constraints (6.27) and (6.28) guarantee

to have control actions always compatible with the energy available in the operator and remote side tanks respectively. As usual with the MPC, the applied command to the robot is just the first value

$$\boldsymbol{u}_o(k) = \hat{\boldsymbol{u}}_o^\star(k). \tag{6.29}$$

6.3.3 Remote side teleoperation

For the remote side, the optimisation problem is similar to the operator side. The state vector $\boldsymbol{z}_r \in \mathbb{R}^N$ has the same dimension of $\boldsymbol{z}_o$ and is given by

$$\boldsymbol{z}_r(k) = \begin{bmatrix} \boldsymbol{q}_o(k-\delta_r(k)) \\ \dot{\boldsymbol{q}}_o(k-\delta_r(k)) \\ E_o(k-\delta_r(k)) \\ \boldsymbol{q}_r(k) \\ \dot{\boldsymbol{q}}_r(k) \\ E_r(k) \\ \hat{\boldsymbol{\tau}}_r(k) \end{bmatrix} \tag{6.30}$$

where $\delta_r(k) \in \mathbb{N}$ is the communication delay from the remote side to the operator side. The equations for f_r and for the cost function $\mathcal{C}_r$ are similar to f_o in (6.24) and $\mathcal{C}_o$ in (6.23), respectively. They should be changed accordingly with the new definition of $\boldsymbol{z}_r$ and with the control input for the remote side robot $\boldsymbol{u}_r$. The optimisation problem

$$\begin{aligned} \hat{\boldsymbol{u}}_r^\star(k+i)|_{i=0}^{k_c} = \quad & \underset{\hat{\boldsymbol{z}},\hat{\boldsymbol{u}}_r}{\arg\min} \quad \mathcal{C}_r(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}_r) \\ & \text{subject to} \quad \hat{\boldsymbol{z}}(k) = \boldsymbol{z}(k) \\ & \qquad\qquad\quad \hat{\boldsymbol{z}}(k+1) = f_r(\hat{\boldsymbol{z}}, \boldsymbol{u}_r) \\ & \qquad\qquad\quad \hat{E}_o \geq \epsilon_o \\ & \qquad\qquad\quad \hat{E}_r \geq \epsilon_r \\ & \qquad\qquad\quad |\hat{\boldsymbol{\tau}}_r| \leq \mathcal{T}^{max} \end{aligned} \tag{6.31}$$

is subjected to an additional constraint (6.31) which takes into account the upper bound $\mathcal{T}^{max}$ on the interaction force.

6.3.4 Passive teleoperation on 1-DOF simulation

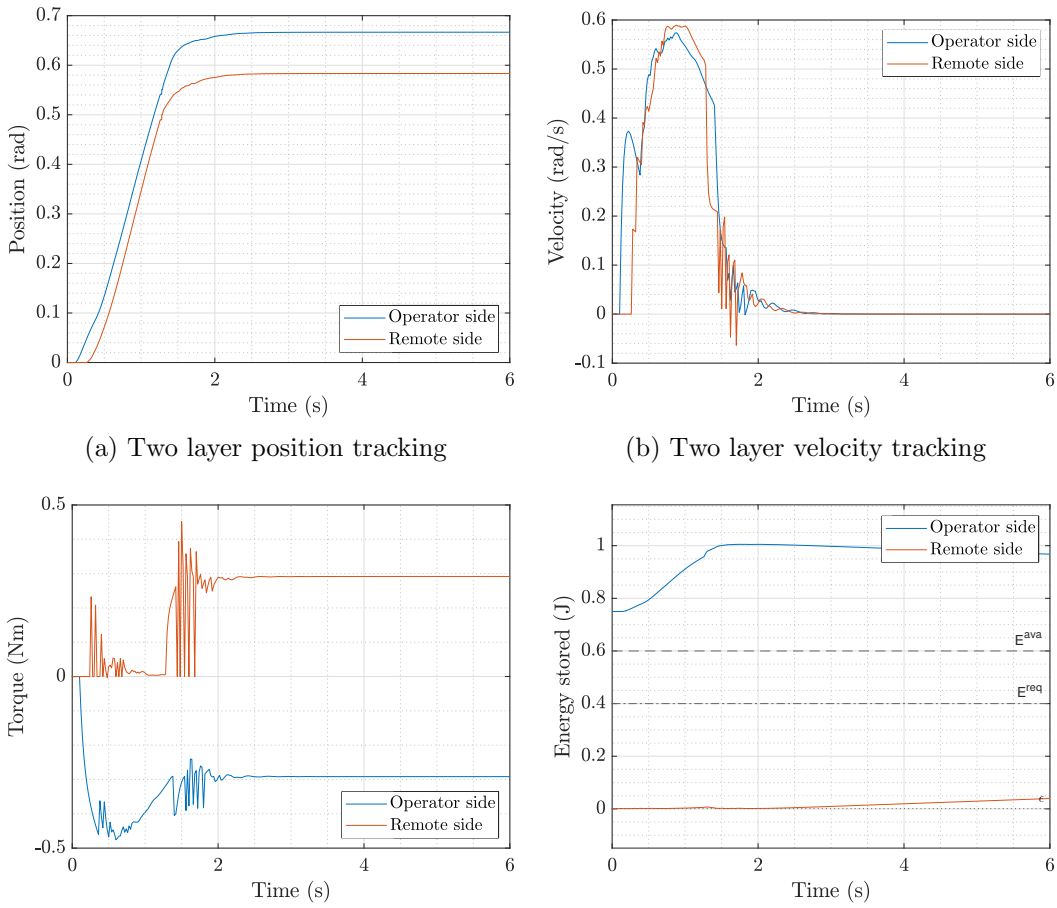
The proposed solution has been initially validated in simulation using *Simulink* and the *Optimisation Toolbox* provided by *Matlab R2019b* on 1 DOF robots. According to Figure 6.1, the robot at the operator side is moved by a PD

controller, acting as the operator, which takes as input the reference signal and the delayed position and velocity of the environment robot; the output is the torque to be applied at the haptic device to track the reference. This feedback loop models the classic video streaming used in a real setup. The P-MPC controllers provide commands both to the operator and environment robots. At the operator side, the command acts as force feedback to the operator, while at the environment side it drives the robot motion. We compared the proposed P-MPC based bilateral teleoperation with the classical *two-layer* architecture proposed in [124] with position PD controllers in the transparency layer. The passivity layer shares the same tank dynamics used to initialize the P-MPC at each control cycle.

The test is conducted with a constant delay of 0.1 s between the operator and the environment side. The sampling time of the controller was set to $\Delta t = 0.02\,\mathrm{s}$, the prediction horizon of the P-MPC equal to 5 steps and the control horizon is 3 steps. The energy thresholds of the two tanks are set equal to small values in order to induce the operator and environment tanks towards the upper and the lower bound respectively ($T^{max} = 1.5$, $T^{ava} = 0.6$, $T^{req} = 0.4$, $\epsilon = 0.001$, $\bar{P} = 0.01$) and in this way to validate the proposed architecture. The tanks are initialised to $T^{max}/2$ at the operator side and to ϵ at the environment side. For this preliminary evaluation, the environment force estimation and the constraint $\mathcal{T}^{max}$ are not taken into account. The robot model parameters (i.e., inertia and friction) used during the simulation are identified on the experimental setup we will use in the Section 6.3.5.

P-MPC and *two-layer* comparison

The experiment is conducted using a low-passed step signal as a position reference for the operator. The two controllers are tuned to behave similarly when the energy stored in the system is enough to perform the action required. We simulate a soft contact placing an obstacle along the path of the environment robot (at 0.5 rad) so it can't reach the desired set point. Figure 6.2 and Figure 6.3 show the original two-layer algorithm (PD controllers) and the P-MPC driving the environment robot to track the position and the velocity of the operator robot until the remote robot interacts with the environment (around 2 s). Figure 6.2c shows a high frequency chattering in the torque commanded to the robot at the environment side: this is because the environment tank has reached its lower bound, as shown in Figures 6.2d and 6.4. Since the passivity layer sends zero torque in case of lack of energy the resulting controller *jitters* around the lower bound until the required energy to apply the torque is



(a) Two layer position tracking

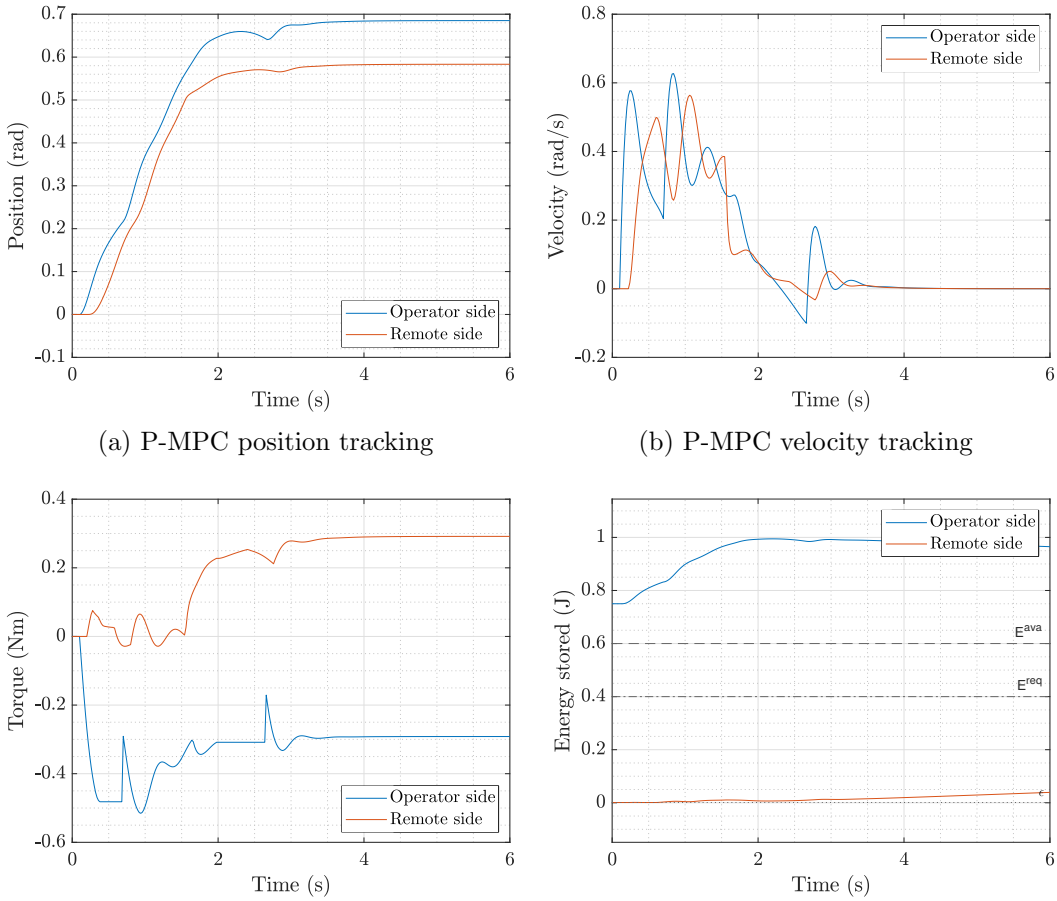
(b) Two layer velocity tracking

(c) Two layer commanded torques

(d) Two layer tanks evolution

Figure 6.2: Two-layers performance during the simulated soft contact. The blue and orange lines are the operator and environment robot trajectories, respectively. a) two-layer position tracking, b) two-layer velocity tracking, c) two-layer torque commanded, d) two-layer stored energy.

transmitted by the operator side tank. On the other hand Figure 6.3c shows a smoother command than Figure 6.2c when the environment tank reaches the lower bound. In fact, as shown in Figure 6.3d and Figure 6.4, the tanks evolution is also smoother since the P-MPC avoids a high rate of change in the command. It's also clear how the MPC avoids the chattering: the position and velocity tracking is less precise when the environment tank reaches the lower bound. In fact, the cost function has to trade-off the tracking error and the rate of change of command. However, in this way, we guarantee a smoother control action (i.e., no vibrations on the motors and reflected to the operator). In Figure 6.2d and Figure 6.3d the increasing of the environment tank at the end of the experiment, starting from 4 s, is due to the incoming power $\bar{P}$ from the operator side. In fact, even if the robots do not move, the environment tank is still requesting energy to the operator side since $E_r < T_r^{req}$



(a) P-MPC position tracking

(b) P-MPC velocity tracking

(c) P-MPC commanded torques

(d) P-MPC tanks evolution

Figure 6.3: P-MPC performance during the simulated soft contact. The blue and orange lines are the operator and environment robot trajectories, respectively. a) P-MPC position tracking, b) P-MPC velocity tracking, c) P-MPC torque commanded, d) P-MPC stored energy.

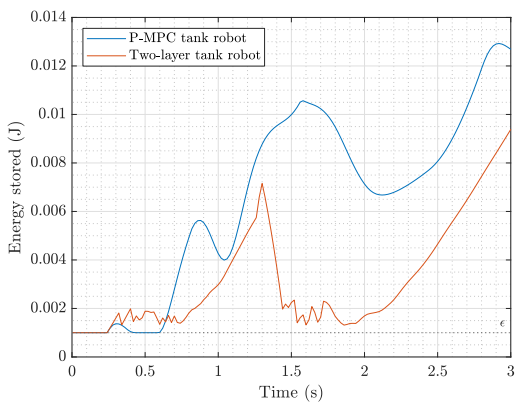


Figure 6.4: A magnification of the environment side tank behaviour during the simulated comparison and the experimental setup. The blue and orange lines are the environment tank evolution for the P-MPC and the two-layer approaches, respectively. The time axis and the data are the same of Figure 6.3 and Figure 6.2.

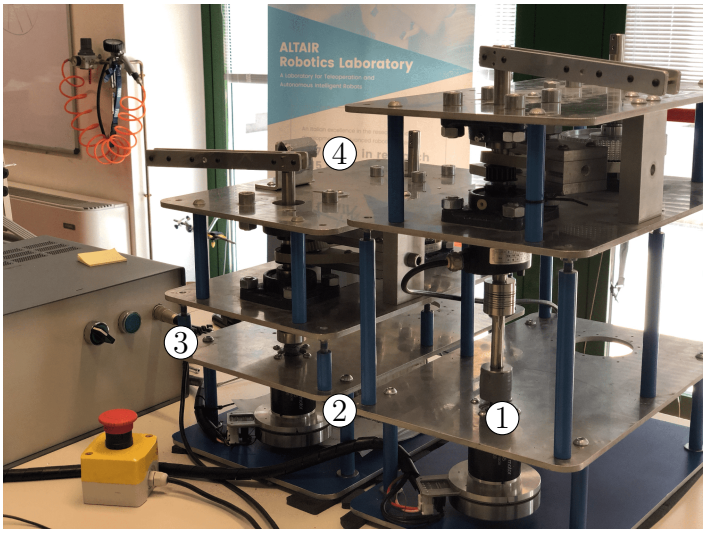


Figure 6.5: The experimental teleoperation setup. (1, 2) DC motors with encoder and gearbox, (3) motors controller, (4) environment side obstacle.

and $E_l \geq T_l^{ava}$.

6.3.5 Passive teleoperation on 1-DOF robot

The P-MPC has been implemented using libmpc++ (see Appendix A) to solve the optimisation problem; the solver is the Sequential Least Squares Programming (SLSQP). The overall teleoperation architecture has been implemented using ROS. The experimental setup is composed of two brushless DC motors controlled via current feedback loops by two *Maxon Escon* boards (see Figure 6.5). The motor's axes are connected through gearboxes and a pair of high precision encoders (4096 readings per rotation) provide position measurements. The operator and environment sides exchange data through a simulated network connection that allows delaying and/or losing packets. The network component queues packets and assigns them a delay value. The communication delays can be constant or randomly time-varying, with the possibility of receiving packets out-of-order. We conducted two experiments, the first with a constant delay of 0.1 s and the second with a uniform random delay between 0.1 and 0.5 s. The sampling time of the controller was set to 0.02 ms, the prediction horizon of the P-MPC is equal to 5 steps and the control horizon is 3 steps. The threshold values used in the two tanks are: $T^{max} = 1.5$, $T^{ava} = 1.0$, $T^{req} = 0.5$, $\epsilon = 0.001$, $\bar{P} = 0.01$. The tanks are initialised to $T^{max}/2$ for the operator side and $T^{req}/2$ for the environment side. As before, the interaction force constraint has not been taken into account. The inertia and the friction of the motor have been identified using a least square approach and the re-

sulting parameters in (6.4) are: $j = 0.0266\,kg\,m^2$ and $b = 0.0218\,N\,s$. The parameters are the same for both motors. Since the experiments focus only on hard contact, the environment stiffness is assumed to be infinite.

Hard contact with constant delay

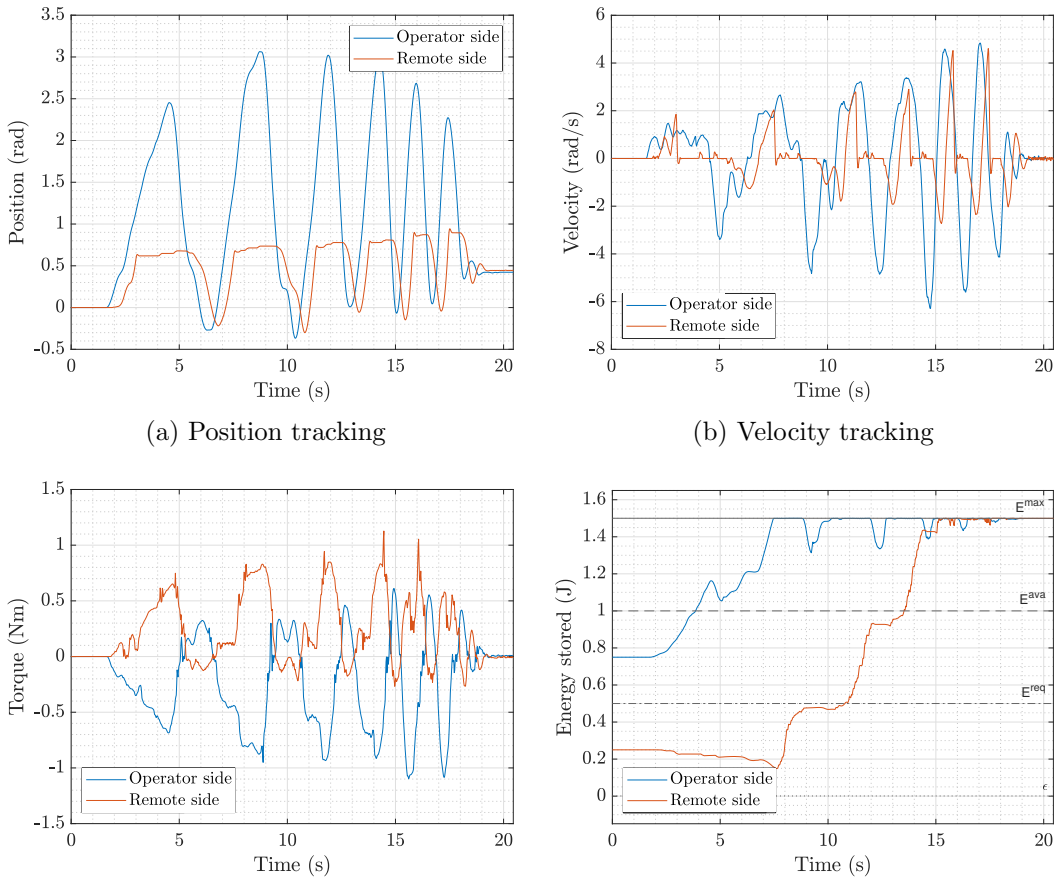
In the first experiment, the operator moves the haptic device pushing the remote robot towards a rigid obstacle placed along its path. The delay is constant and as it can be seen in Figures 6.6a and 6.6b the remote robot goes in contact several times with the environment. The contact position moves a bit during the experiments and this is due to a backlash in the handle attached to the motor. Figure 6.6c shows the command computed by the P-MPCs: the operator and the environment torques match during the contacts (with opposite signs). Figure 6.6d shows the evolution of the tanks, and they are correctly upper-bounded and the action of the new P^{out} definition can be seen between 7 and 9 s. In that interval the operator tank is full and the power dissipated by the operator is not "wasted" but transferred to the environment side. This causes a faster increase of the available energy for the environment robot allowing a more reactive response.

Hard contact with variable delay

In the second scenario, the communication delay changes randomly over time. The operator moves the robot and the motion at the environment side is blocked by an obstacle. Figure 6.7c shows the command torques: also in this case the interaction with the environment is correctly reflected to the operator. The tanks are initialised as in the previous experiment and as before between 10 and 13 s the environment tank receives the dissipated power by the operator which guarantees a faster response. Figure 6.7a shows a small movement of the contact position due to the backlash in the robot handle.

6.3.6 Force constraint on 6-DOF simulation

The proposed P-MPC control law has been validated in a simulated environment using *Simulink* and the *Optimisation Toolbox* provided by *Matlab 2021a*. Figure 6.8 shows the two 6-DOF UR5 robots used at both sides of the teleoperation system. The control architecture is composed of two controllers: an inverse dynamic controller and the teleoperation controller. The inner loop performs gravity compensation and dynamic linearisation. The resulting system is then torque controlled by the teleoperation architecture. The action



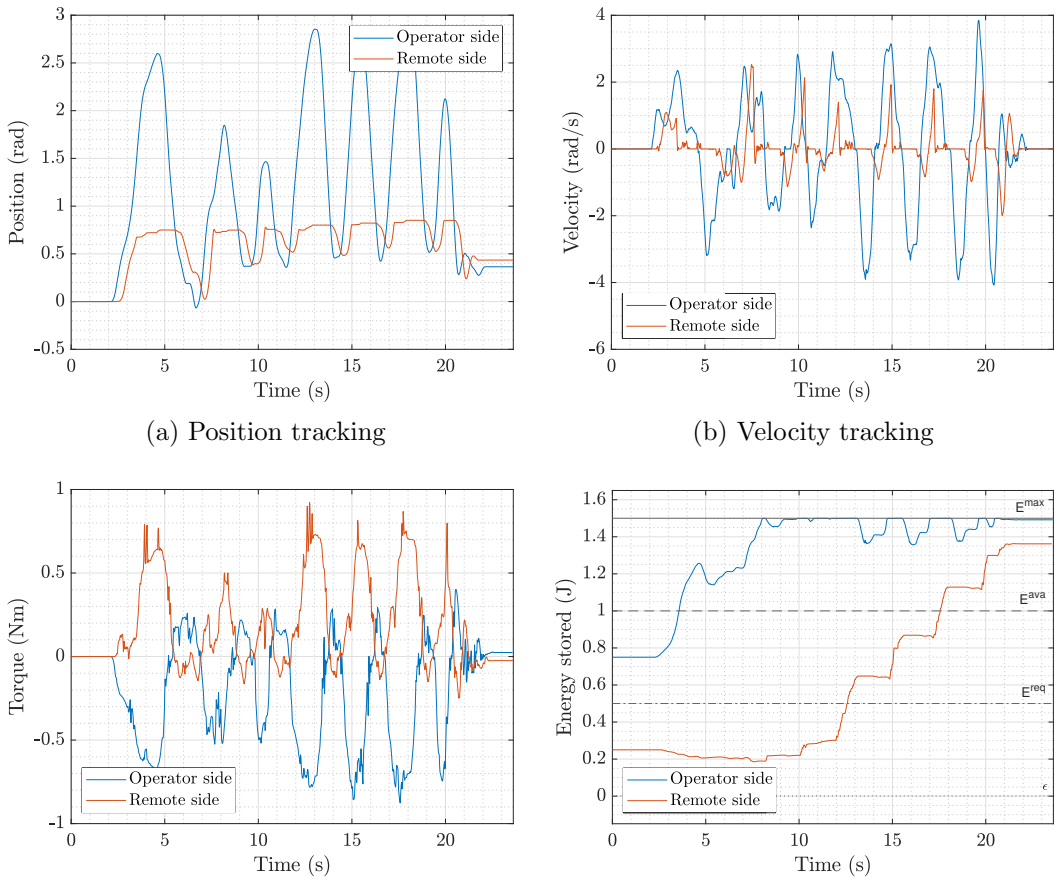
(a) Position tracking

(b) Velocity tracking

(c) Commanded torques

(d) Tanks evolution

Figure 6.6: Hard contact with constant delay P-MPC. The blue and orange lines are the operator and environment robot trajectories respectively. a) position tracking, b) velocity tracking, c) torque commanded, d) stored energy.



(a) Position tracking

(b) Velocity tracking

(c) Commanded torques

(d) Tanks evolution

Figure 6.7: Hard contact with variable delay P-MPC. The blue and orange lines are the operator and environment robt trajectories respectively. a) position tracking, b) velocity tracking, c) torque commanded, d) stored energy.

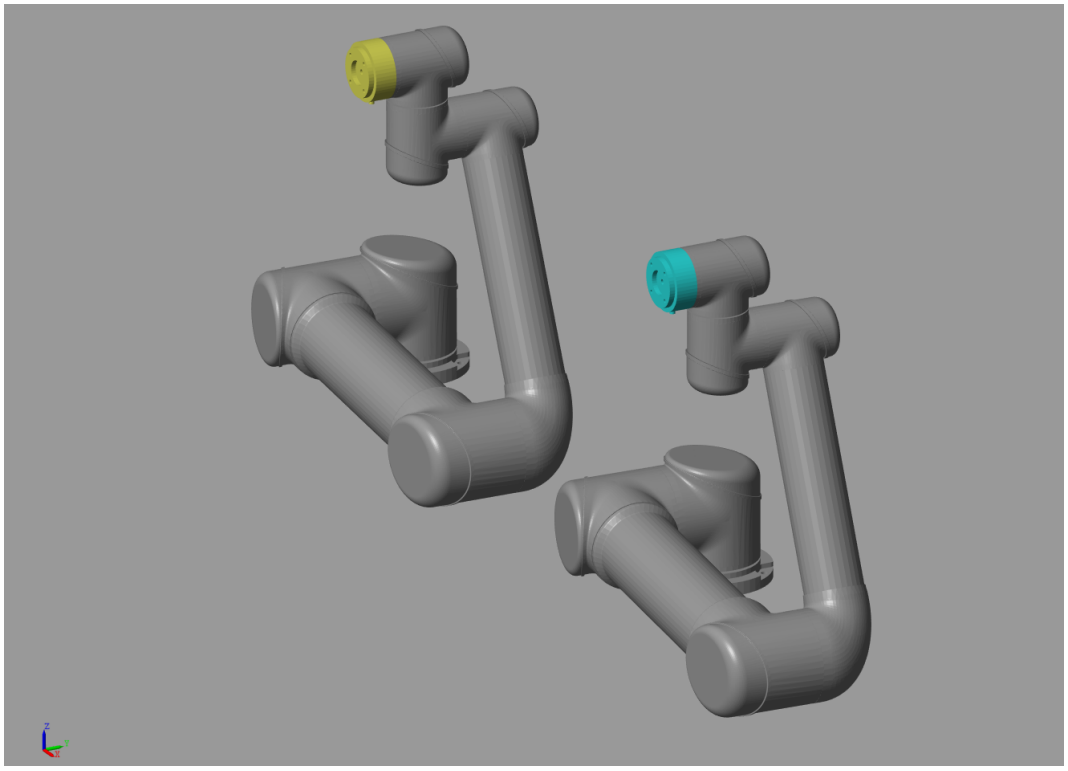


Figure 6.8: Simulation setup. The blue and the yellow ring shows the operator and remote side robot's end-effector, respectively.

of the operator (i.e., $\boldsymbol{\tau}_o$) is modelled as a PD position controller between a Cartesian reference trajectory and the remote side robot Cartesian position. This feedback loop models the behaviour of the operator watching the delayed video streaming in real teleoperation setups. The output of this feedback is the force/torque applied to the haptic device. The two P-MPC controllers provide commands to the operator and the remote side robots. At the operator side, the command acts as force feedback to the operator, while at the environment side it controls the motion of the robot.

The experiments are conducted assuming to have partial prior knowledge on the environment model. The nominal parameters are within lower bounds and upper bounds which are application-dependant and can be known, for example, by previous interactions. From that, we take as initial guess an overestimation of the damping and the stiffness parameters of the model. This choice, despite a more conservative control action, guarantees the satisfaction of the constraints also during the convergence of the estimator. The *true* parameters of the environment model are the following

$$\boldsymbol{K}_e = \mathrm{diag}(0, 80, 0, 0, 0, 0)$$
$$\boldsymbol{B}_e = \mathrm{diag}(1.5, 7, 1.5, 0, 0, 0)$$

where we assume to interact with a spring along the y axis in a viscous environment. The environment estimator is set up with an initial guess of $\boldsymbol{K}_e = \text{diag}(0, 88, 0, 0, 0, 0)$ and $\boldsymbol{B}_e = \text{diag}(1.65, 7.7, 1.65, 0, 0, 0)$, which are a 10% overestimation of the real values. The inertia and the viscous friction of the linearised dynamic model (6.5) are the same for both the teleoperation side and equal to

$$\boldsymbol{J} = \text{diag}(0.1, 0.1, 0.1, 0.1, 0.1, 0.1)$$
$$\boldsymbol{B} = \text{diag}(0.4, 0.4, 0.4, 0.04, 0.4, 0.4)$$

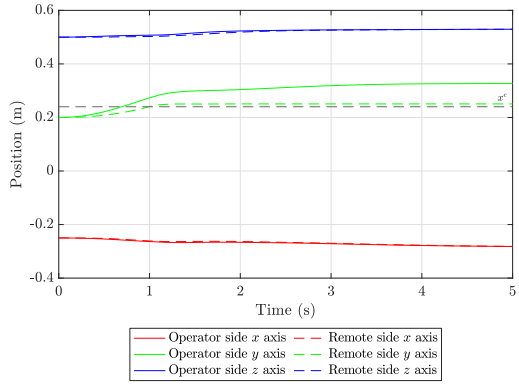
which results in a set of stable second-order linear systems. The sampling time of the controller was set to $\Delta t = 0.02\,\text{s}$, the prediction horizon k_p of the P-MPC is equal to 5 steps while the control horizon k_c is 3 steps. The thresholds of the tanks are the same at both sides of the teleoperation system and are set to $T^{max} = 1$, $T^{ava} = 0.6$, $T^{req} = 0.2$, $\epsilon = 0.001$, $\bar{P} = 0.01$. The tanks are initialised to $T^{max}/2$ for the operator side and $T^{max}/2$ for the remote side.

Contact with no communication delay

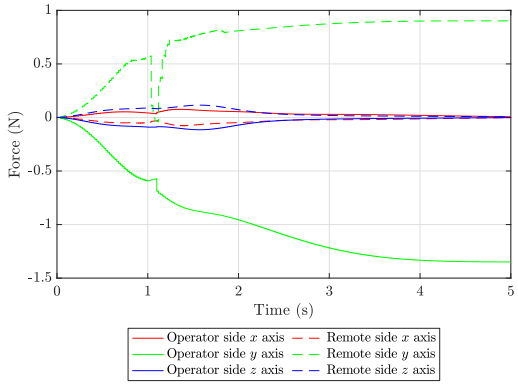
In this first scenario, the operator side robot is moved along a filtered step trajectory on the x, y and z axes. A soft obstacle is placed ahead of the starting position of the remote side robot. The maximum interaction force $\mathcal{T}^{max}$ is set to 0.85 N which is the 75% of the peak force obtained without such constraint. The communication delay is set to zero. Figure 6.9a shows the Cartesian position of the operator and remote side robots. Figure 6.9b shows the control action computed by the P-MPC. It is possible to see how the force feedback provided to the operator matches the environment torque at the equilibrium. Figure 6.9e shows the behaviour of the interaction force with the environment. As expected, the P-MPC is able to limit the interaction force below the upper threshold. Figure 6.9d and Figure 6.9c show the environment parameters estimation: the estimator converges to the expected stiffness and damping values. Finally, Figure 6.9f shows the evolution of the energy tanks.

Contact with constant communication delay

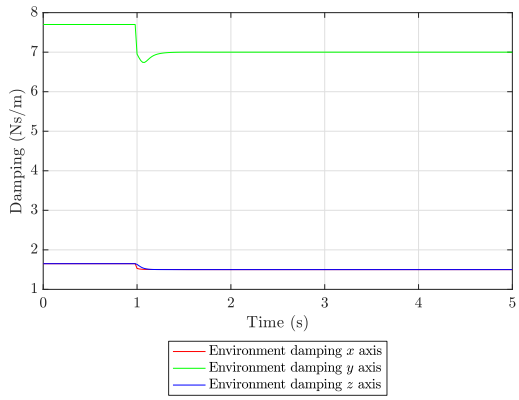
In the second scenario we tested our method with a round-trip communication delay of 150 ms, $\delta_o(k) = 75\,\text{ms}$ and $\delta_r(k) = 75\,\text{ms}$. The operator side robot is moved again along the y direction and as before an obstacle is placed ahead of the starting position of the remote side robot. In this case the maximum interaction force $\mathcal{T}^{max}$ results in 0.55 N which is again the 75% of the peak



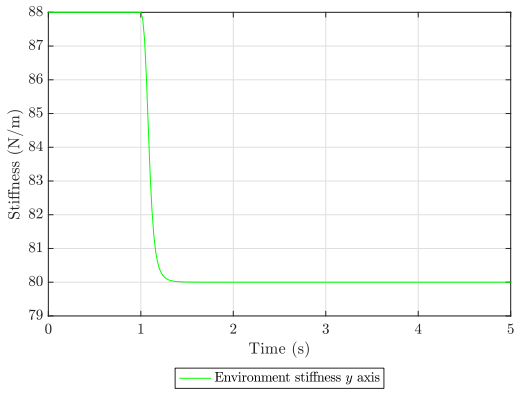
(a) Position coupling



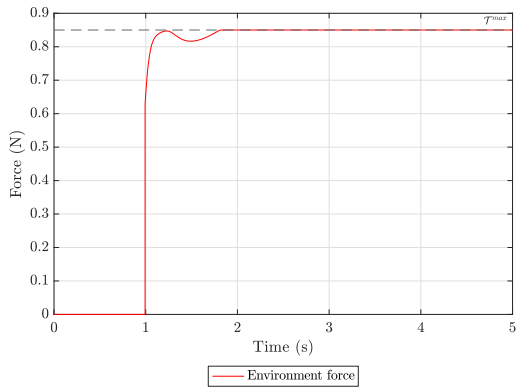
(b) Command torque



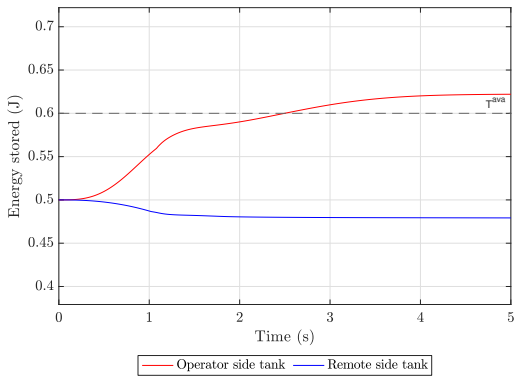
(c) Estimated environment damping



(d) Estimated environment stiffness

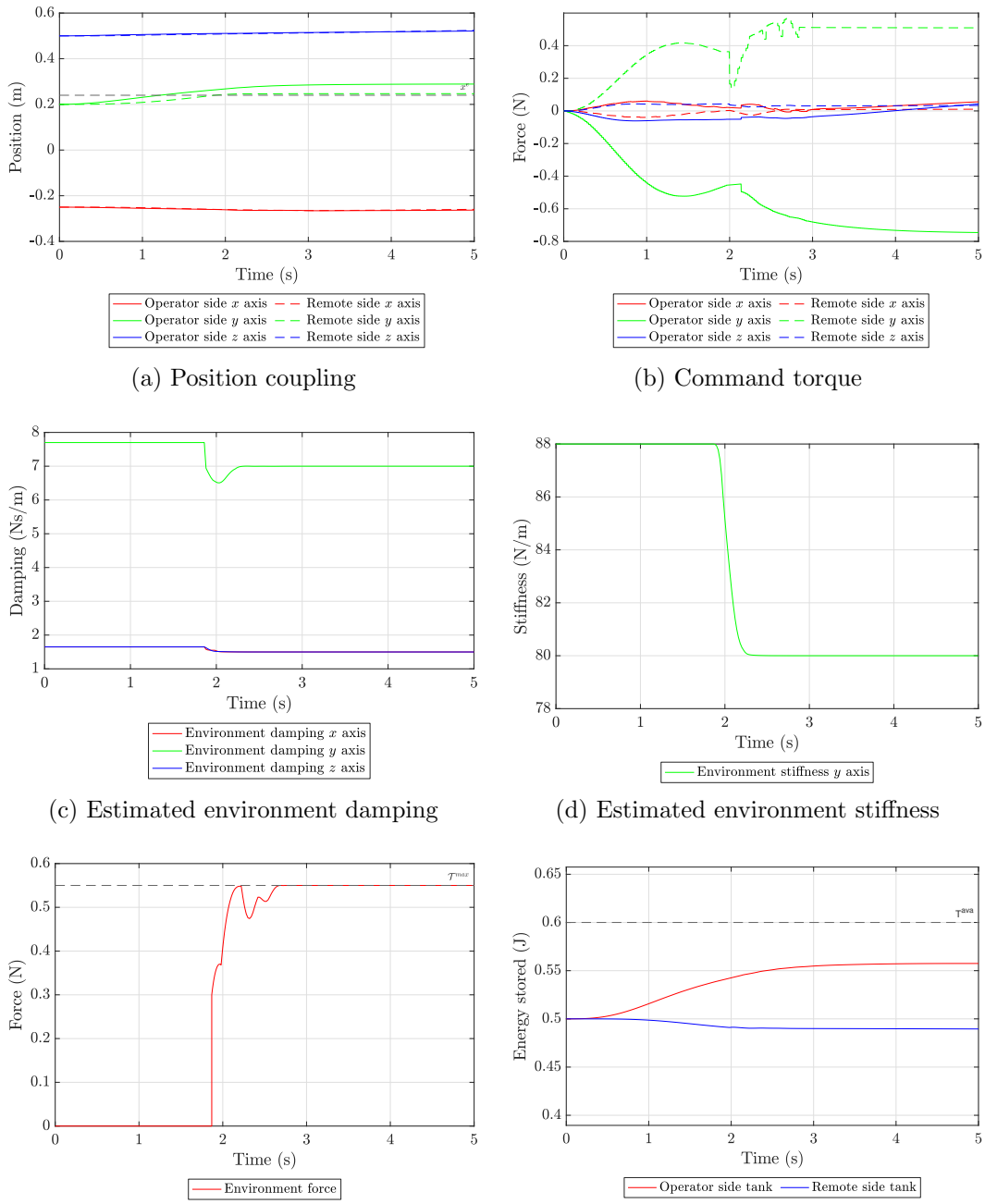


(e) Modulus of the interaction torque



(f) Energy tanks

Figure 6.9: Simulation results for the soft contact scenario with no communication delay. a) Cartesian position along x, y, z axes (red, green and blue lines respectively) of the operator (solid line) and remote (dashed line) side robot, b) command torque along x, y, z axes of the operator and remote side robot, c) estimation of environment damping along x, y, z axes, d) estimation of the environment stiffness along the y axis, e) modulus of the interaction force of the remote side robot, d) energy tanks at the operator and remote sides.



(a) Position coupling

(b) Command torque

(c) Estimated environment damping

(d) Estimated environment stiffness

(e) Modulus of the interaction torque

(f) Energy tanks

Figure 6.10: Simulation results for the soft contact scenario with constant communication delay. a) Cartesian position along x, y, z axes (red, green and blue lines respectively) of the operator (solid line) and remote (dashed line) side robot, b) command torque along x, y, z axes of the operator and remote side robot, c) estimation of environment damping along x, y, z axes, d) estimation of the environment stiffness along the y axis, e) modulus of the interaction force of the remote side robot, d) energy tanks of the operator and remote side robot.

force without the constraint. The parameters of the environment model and the initial guess of the estimator are the same as in the previous experiment.

The position tracking of the operator side and remote side robots is shown in Figure 6.10a. The P-MPC control actions, shown in Figure 6.10b, is able to provide a force feedback to the operator that matches the environment torque when the system reaches the equilibrium. Moreover, the P-MPC controller guarantees the smoothness of the command torques also in case of communication delay. Regarding the interaction force behaviour, as shown in Figure 6.10e, also in this case the P-MPC limits the environment torque to the upper bound. Thanks to the overestimation of the environment parameters, the torque is bounded also during the convergence of the estimator, as shown in Figure 6.10d and Figure 6.10c. The convergence of the parameters to the expected values takes 0.5 s. Finally, Figure 6.10f shows the evolution of the energy tanks.

6.4 Force constraint using adaptive linear MPC

In the previous section we defined and implement the bilateral teleoperation solving a non-linear optimization problem. And even though the proposed methodology is effective it cannot be implemented for real-time control of multi-DOF manipulators due to the high computational effort. The solution proposed to exploit such technique also for real-time control, as mentioned before, is the design of an adaptive linear P-MPC.

Adaptive control is a special kind of non-linear control in which the controller has adjustable parameters with an adjustment law. As opposed to robust control, adaptive control does not need any prior information about the uncertainties' bounds. In fact, instead of providing stable control law within given bounds, adaptive control schemes deal with changing control law. An adaptive control system can be considered made of two loops. The main loop is in general classic feedback with the plant and the controller, while the other one is the loop containing parameters adaption. Usually, the loop containing the parameter adjustment is slower with respect to the main loop.

The first step in this direction is the adoption of an MPC to constraint the interaction forces while performing industrial manipulation tasks. Then we will extend this control architecture with a linear approximation of the tank dynamics to model the whole bilateral teleoperation system and define the proper MPC controller. This solution takes explicitly into account the interaction force as a part of the output vector by updating the system's model

accordingly to the contact state of the manipulator. The linearized model used within the MPC is defined in the operational space to remove the dependence from the manipulator's kinematics and to allow an easy integration between the environment and robot models. The linearization is possible thanks to the identification of the robot's dynamic parameters and the control law implemented in the robot's driver (that is not possible to modify or overcome). We resort to the Recursive Least Square (RLS) approach as proposed in [169] to perform both the robot's dynamic parameters estimation and the MPC controller parameters adjustment.

The Euler-Lagrange model for a manipulator is defined as

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \underbrace{\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}_v\dot{\boldsymbol{q}} + \boldsymbol{F}_s sgn(\dot{\boldsymbol{q}})}_{\boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}})} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}_e + \boldsymbol{\tau} \tag{6.32}$$

where $\boldsymbol{q} \in \mathbb{R}^m$ are the generalized coordinates of the robot, $\boldsymbol{M}(\boldsymbol{q})$, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$, $\boldsymbol{g}(\boldsymbol{q})$ are the inertia, Coriolis and gravity terms, $\boldsymbol{F}_v$ and $\boldsymbol{F}_s$ are the viscous and static friction, $\boldsymbol{\tau}_e$ and $\boldsymbol{\tau}$ are the external and the control torques. Assuming the interaction between the manipulator and the environment happens only at the TCP (Tool Center Point) the resulting wrench $\boldsymbol{h}_e$ can be projected into the corresponding joint torques by using the robot TCP Jacobian $\boldsymbol{J}(\boldsymbol{q})$ as

$$\boldsymbol{\tau}_e = \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{h}_e. \tag{6.33}$$

The motor torques $\boldsymbol{\tau}$ are usually not directly controllable, especially in case of industrial manipulators. For instance, in case of inner position controller its input variables are the desired configuration $\boldsymbol{q}_r$ and the current position/velocity, while the outputs are the control torques

$$\boldsymbol{\tau} = \kappa(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_r). \tag{6.34}$$

Substituting (6.34) in (6.32)

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}_e + \kappa(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_r) \tag{6.35}$$

we get the closed-loop dynamics where the *new control input* is $\boldsymbol{q}_r$. For this kind of manipulator, the inverse dynamics control law [164] can be applied only if the dynamics of the manipulator *and* the controller can be inverted at the same time since we cannot act directly on $\boldsymbol{\tau}$.

Robot identification

In case the manipulator dynamic model is not available it can be identified by using regression techniques as in [169]–[171]. The joint torques can be

computed exploiting the linear-in-the-dynamic-parameters property of serial link manipulators as

$$\begin{bmatrix} \tau_1 \\ \tau_2 \\ \vdots \\ \tau_m \end{bmatrix} = \begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1m} \\ 0 & y_{22} & \cdots & y_{2m} \\ \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & y_{mm} \end{pmatrix} \begin{bmatrix} \boldsymbol{\theta}_1 \\ \boldsymbol{\theta}_2 \\ \vdots \\ \boldsymbol{\theta}_m \end{bmatrix} \tag{6.36}$$

where $\boldsymbol{Y} \in \mathbb{R}^{m \times mk}$ is the regression matrix and $\boldsymbol{\theta}_i \in \mathbb{R}^k$ is the vector containing the parameters of the i-th link. In our case, the parameters vector is

$$\boldsymbol{\theta}_i = \begin{bmatrix} m_i & m_i \boldsymbol{r}_{i,C_i}^i & \hat{\boldsymbol{I}}_i^i & \boldsymbol{I}_{m_i} & \boldsymbol{F}_{v_i}^{\pm} & \boldsymbol{F}_{s_i}^{\pm} \end{bmatrix}^T \tag{6.37}$$

where m_i is the mass of the link, $\boldsymbol{r}_{i,C_i} \in \mathbb{R}^3$ is the center of mass, $\hat{\boldsymbol{I}}_i^i \in \mathbb{R}^6$ collects the six independent entries in the symmetric inertia matrix, $\boldsymbol{I}_{m_i}$ is the actuator inertia along the rotational axis, $\boldsymbol{F}_{v_i}^{\pm}$ and $\boldsymbol{F}_{s_i}^{\pm}$ are the viscous and static frictions of the i-th actuator, respectively (two values each representing the two different directions of motion). Considering N instants of time we can define

$$\begin{bmatrix} \boldsymbol{\tau}_1 \\ \boldsymbol{\tau}_2 \\ \vdots \\ \boldsymbol{\tau}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{Y}(\boldsymbol{q}_1, \dot{\boldsymbol{q}}_1, \ddot{\boldsymbol{q}}_1) \\ \boldsymbol{Y}(\boldsymbol{q}_2, \dot{\boldsymbol{q}}_2, \ddot{\boldsymbol{q}}_2) \\ \vdots \\ \boldsymbol{Y}(\boldsymbol{q}_N, \dot{\boldsymbol{q}}_N, \ddot{\boldsymbol{q}}_N) \end{bmatrix} \boldsymbol{\theta} \tag{6.38}$$

where $\boldsymbol{\tau}_i \in \mathbb{R}^m$ and $\boldsymbol{Y}(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i, \ddot{\boldsymbol{q}}_i)$ are the control torques vector and the regression matrix at the i-th instant of time; we recall [164], [169] for the computation of $\boldsymbol{Y}(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i, \ddot{\boldsymbol{q}}_i)$. The estimation of $\boldsymbol{\theta}$ is obtained by solving the linear system (6.38).

Robot controller identification

The inner controller of the UR5e robot is assumed to be a PD position controller in the joint space with gravity compensation like [164]

$$\boldsymbol{\tau} = \kappa(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{q}_r) = \boldsymbol{K}_p(\boldsymbol{q}_r - \boldsymbol{q}) + \boldsymbol{K}_d(\dot{\boldsymbol{q}}_r - \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) \tag{6.39}$$

where $\boldsymbol{K}_p$ and $\boldsymbol{K}_d$ are the positive definite proportional and derivative gain matrices. Their identification is performed again with a least squares approach where the input-output data $\{\boldsymbol{q}(t_i), \dot{\boldsymbol{q}}(t_i), \boldsymbol{q}_r(t_i), \dot{\boldsymbol{q}}_r(t_i); \boldsymbol{\tau}(t_i)\}$ are obtained by moving the robot in free-motion. The identification of the controller is performed after the identification of the manipulator dynamics in order to remove the gravity term $\boldsymbol{g}(\boldsymbol{q})$ from (6.39). Even if the controller structure assumption

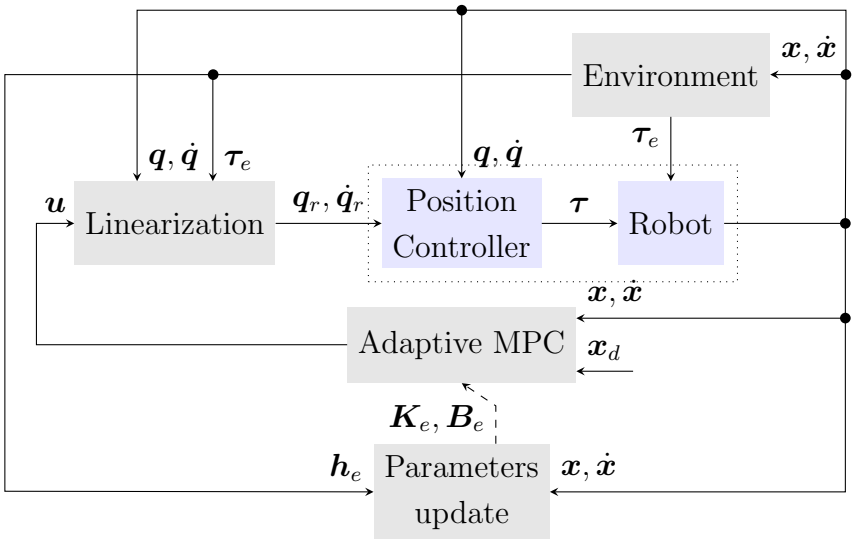


Figure 6.11: The block diagram of the proposed control architecture; in light blue the manipulator dynamics and the inner joints controller. The control torque $\boldsymbol{\tau}$ within the dotted box is exclusively managed by the robot driver.

has no finding in the literature, the experimental validation of the parameter estimation shows that such it should be correct.

Substituting (6.39) in (6.35)

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau}_e + \boldsymbol{K}_p(\boldsymbol{q}_r - \boldsymbol{q}) - \boldsymbol{K}_d(\dot{\boldsymbol{q}}_r - \dot{\boldsymbol{q}}) \tag{6.40}$$

we can derive the closed-loop dynamics of the manipulator together with the internal controller

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{K}_p\boldsymbol{q} + \boldsymbol{K}_d\dot{\boldsymbol{q}} = \boldsymbol{\tau}_e + \boldsymbol{K}_p\boldsymbol{q}_r + \boldsymbol{K}_d\dot{\boldsymbol{q}}_r. \tag{6.41}$$

6.4.1 MPC with force constraint

As mentioned above, the proposed control architecture allows to take into account the constraint on the interaction forces with the environment even if the manipulator is not controlled with a torque control law. The block diagram of our architecture is shown in Figure 6.11 where the feedback linearization is exploited to use a linear model within the MPC to reduce its computational burden. For the sake of simplicity, in the following formulation we do not consider explicitly the friction terms. Let

$$\bar{\boldsymbol{M}}\ddot{\boldsymbol{x}} + \bar{\boldsymbol{D}}\dot{\boldsymbol{x}} = \boldsymbol{u}. \tag{6.42}$$

be the aforementioned linearized model where $\ddot{\boldsymbol{x}}$, $\dot{\boldsymbol{x}}$ are the operational space acceleration and velocity and $\boldsymbol{u}$ is the command wrench computed by the MPC.

The matrices $\bar{\boldsymbol{M}}$ and $\bar{\boldsymbol{D}}$ are the desired inertia and damping. The exact state feedback linearization in the operational space as in [164] of (6.41) with respect to the control inputs $\boldsymbol{q}_r$ and $\dot{\boldsymbol{q}}_r$ is defined by the following first-order ODE

$$\boldsymbol{K}_p\boldsymbol{q}_r + \boldsymbol{K}_d\dot{\boldsymbol{q}}_r = \boldsymbol{M}(\boldsymbol{q})\boldsymbol{y} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{K}_p\boldsymbol{q} + \boldsymbol{K}_d\dot{\boldsymbol{q}} - \boldsymbol{\tau}_e \tag{6.43}$$

such that by substituting (6.43) in (6.41) we obtain

$$\ddot{\boldsymbol{q}} = \boldsymbol{y} \tag{6.44}$$

where the new input $\boldsymbol{y}$ is defined as

$$\boldsymbol{y} = \boldsymbol{J}^{-1}(\boldsymbol{q})(\ddot{\boldsymbol{x}} - \dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}}). \tag{6.45}$$

Substituting (6.42) in (6.45)

$$\boldsymbol{y} = \boldsymbol{J}^{-1}(\boldsymbol{q})(\bar{\boldsymbol{M}}^{-1}(\boldsymbol{u} - \bar{\boldsymbol{D}}\dot{\boldsymbol{x}}) - \dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}}) \tag{6.46}$$

we can derive the equations of the non-linear feedback by substituting (6.46) in (6.43)

$$\begin{aligned}\boldsymbol{K}_p\boldsymbol{q}_r + \boldsymbol{K}_d\dot{\boldsymbol{q}}_r = \boldsymbol{M}(\boldsymbol{q})(\boldsymbol{J}^{-1}(\boldsymbol{q})((\bar{\boldsymbol{M}}^{-1}(\boldsymbol{u} - \bar{\boldsymbol{D}}\dot{\boldsymbol{x}}) - \dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}})) + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ + \boldsymbol{K}_p\boldsymbol{q} + \boldsymbol{K}_d\dot{\boldsymbol{q}} - \boldsymbol{\tau}_e\end{aligned} \tag{6.47}$$

Finally, the expression in (6.47) can be rewritten in terms of the identified manipulator dynamics (6.36) and the measured wrench $\boldsymbol{h}_e$ as

$$\boldsymbol{K}_p\boldsymbol{q}_r + \boldsymbol{K}_d\dot{\boldsymbol{q}}_r = \boldsymbol{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{y})\boldsymbol{\theta} - \boldsymbol{Y}(\boldsymbol{q}, \boldsymbol{0}, \boldsymbol{0})\boldsymbol{\theta} + \boldsymbol{K}_p\boldsymbol{q} + \boldsymbol{K}_d\dot{\boldsymbol{q}} - \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{h}_e \tag{6.48}$$

It is worth remarking that by applying (6.48) to (6.41) the resulting closed-loop dynamics is linear in the Cartesian space and the external force/torque are compensated. Moreover, since the system (6.42) is controlled by the wrench command $\boldsymbol{u}$, the feedback linearization loop acts as an admittance controller. Consequently, the MPC acts as an impedance controller with input $\boldsymbol{x}_d$ and output $\boldsymbol{u}$.

Adaptive model predictive control

Let $\boldsymbol{z}$ be the state vector of the system modelled within the MPC

$$\boldsymbol{z} = \begin{bmatrix} \boldsymbol{x} \\ \dot{\boldsymbol{x}} \end{bmatrix} \tag{6.49}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\dot{\boldsymbol{x}} \in \mathbb{R}^n$ are the Cartesian position and velocity of the end-effector. The state-space representation of (6.42) and the environment model (6.13) together with the measurement equation is

$$\begin{cases} \dot{\boldsymbol{z}}(t) = \boldsymbol{A}\boldsymbol{z}(t) + \boldsymbol{B}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{z}(t) + \boldsymbol{D}^d\boldsymbol{d}(t) \end{cases} \tag{6.50}$$

where the signal

$$\boldsymbol{d}(t) = \sigma(t)\boldsymbol{x}_e \tag{6.51}$$

is the position of environment if $\sigma(t) = 1$ or zero otherwise. The state evolution matrix $\boldsymbol{A} \in \mathbb{R}^{2n\times 2n}$ is

$$\boldsymbol{A} = \begin{pmatrix} 0^{n\times n} & \boldsymbol{I}^{n\times n} \\ 0^{n\times n} & -\boldsymbol{M}^{-1}\boldsymbol{D} \end{pmatrix} \tag{6.52}$$

while the input matrix $\boldsymbol{B} \in \mathbb{R}^{2n\times n}$ is

$$\boldsymbol{B} = \begin{pmatrix} 0^{n\times n} \\ \boldsymbol{M}^{-1} \end{pmatrix}. \tag{6.53}$$

The output matrix $\boldsymbol{C}$ is time-varying since it depends on the contact state of the manipulator with the environment: the robot model switches between two matrices of dimension $2n \times 2n$ according to $\sigma(t)$ as follows

$$\boldsymbol{C} = \begin{cases} \begin{pmatrix} \boldsymbol{I}^{n\times n} & 0^{n\times n} \\ 0^{n\times n} & 0^{n\times n} \end{pmatrix}, & \text{if } \sigma(t) = 0 \\ \\ \begin{pmatrix} \boldsymbol{I}^{n\times n} & 0^{n\times n} \\ -\boldsymbol{K}_e & -\boldsymbol{B}_e \end{pmatrix}, & \text{if } \sigma(t) = 1 \end{cases}. \tag{6.54}$$

Finally, the output disturbance matrix $\boldsymbol{D}^d \in \mathbb{R}^{2n\times n}$ is defined as

$$\boldsymbol{D}^d = \begin{pmatrix} 0^{n\times n} \\ \boldsymbol{K}_e \end{pmatrix}. \tag{6.55}$$

The output vector of the adaptive MPC

$$\boldsymbol{y}(t) = \begin{bmatrix} \hat{\boldsymbol{x}}(t) \\ \hat{\boldsymbol{h}}_e(t) \end{bmatrix}$$

consists of the Cartesian pose and the interaction force according to the model (6.13). The measurements of the real interaction force with the environment

are read thanks to the force-torque sensor at the end-effector of the manipulator. The vector $\boldsymbol{y}$ will be used in the cost function of the optimization problem to penalize the tracking error ($\boldsymbol{e} = \boldsymbol{x}_d - \boldsymbol{x}$) and to constrain the maximum interaction force ($|\hat{\boldsymbol{h}}_e| \leq \mathcal{T}^{max}$). We also assume to know the velocity $\dot{\boldsymbol{x}}$ thanks to the values of $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ provided by the robot driver.

The discrete-time model used in the MPC is the following

$$\begin{cases} \boldsymbol{z}(k+1) = f(\boldsymbol{z}(k), \boldsymbol{u}(k), \boldsymbol{d}(k)) = \bar{\boldsymbol{A}}\boldsymbol{z}(k) + \bar{\boldsymbol{B}}\boldsymbol{u}(k) \\ \boldsymbol{y}(k) = \bar{\boldsymbol{C}}\boldsymbol{z}(k) + \bar{\boldsymbol{D}}^d\boldsymbol{d}(k) \end{cases} \tag{6.56}$$

where the matrices $\bar{\boldsymbol{A}}$, $\bar{\boldsymbol{B}}$, $\bar{\boldsymbol{C}}$ and $\bar{\boldsymbol{D}}^d$ are the discrete-time equivalent of the continuous dynamics previously described and k denotes $t_k = k\Delta t$. The cost function to be minimised would balance the position tracking error between the operator pose and the reference trajectory $\boldsymbol{x}_d$, the control input $\hat{\boldsymbol{u}}$ and its rate of change $\Delta\hat{\boldsymbol{u}}$. The resulting quadratic cost function at time k is

$$\begin{aligned} \mathcal{C}(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}) = & \sum_{i=1}^{k_p} \hat{\boldsymbol{e}}^T(k+i)\boldsymbol{\Lambda}_p\hat{\boldsymbol{e}}(k+i) + \\ & + \sum_{j=1}^{k_c} \Big[\hat{\boldsymbol{u}}^T(k+j)\boldsymbol{\Lambda}_c\hat{\boldsymbol{u}}(k+j) \\ & + \Delta\hat{\boldsymbol{u}}^T(k+j)\boldsymbol{\Lambda}_d\Delta\hat{\boldsymbol{u}}(k+j)\Big] \end{aligned} \tag{6.57}$$

where

$$\hat{\boldsymbol{e}}(k+i) \triangleq \hat{\boldsymbol{x}}(k+i) - \hat{\boldsymbol{x}}_d(k+i), \tag{6.58}$$

which is the tracking error computed along the prediction horizon. $\boldsymbol{\Lambda}_p$, $\boldsymbol{\Lambda}_c$ and $\boldsymbol{\Lambda}_d$ are application-dependant positive definite weighting matrices, k_p and k_c are the prediction and control horizons. The optimal control commands $\hat{\boldsymbol{u}}^\star(k) \ldots \hat{\boldsymbol{u}}^\star(k+k_c)$ are obtained as a solution of the finite-horizon constrained optimal LQ control problem

$$\begin{aligned} \hat{\boldsymbol{u}}^\star(k+i)|_{i=0}^{k_c} = \quad & \arg\min_{\hat{\boldsymbol{u}}(\cdot)} \quad \mathcal{C}(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}, \boldsymbol{d}) \\ & \text{subject to} \quad \hat{\boldsymbol{z}}(k) = \boldsymbol{z}(k) \\ & \qquad\qquad\quad \hat{\boldsymbol{z}}(k+1) = f(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}, \boldsymbol{d}) \\ & \qquad\qquad\quad \hat{\boldsymbol{h}}_e \leq \mathcal{T}^{max} \qquad (6.59) \\ & \qquad\qquad\quad \hat{\boldsymbol{h}}_e \geq 0 \qquad (6.60) \end{aligned}$$

where the estimated interaction force $\hat{\boldsymbol{h}}_e$ within the optimization horizon is computed as

$$\hat{\boldsymbol{h}}_e = \hat{\boldsymbol{K}}_e(\hat{\boldsymbol{x}}_e - \hat{\boldsymbol{x}}(k)) - \hat{\boldsymbol{B}}_e\dot{\hat{\boldsymbol{x}}}(k). \tag{6.61}$$

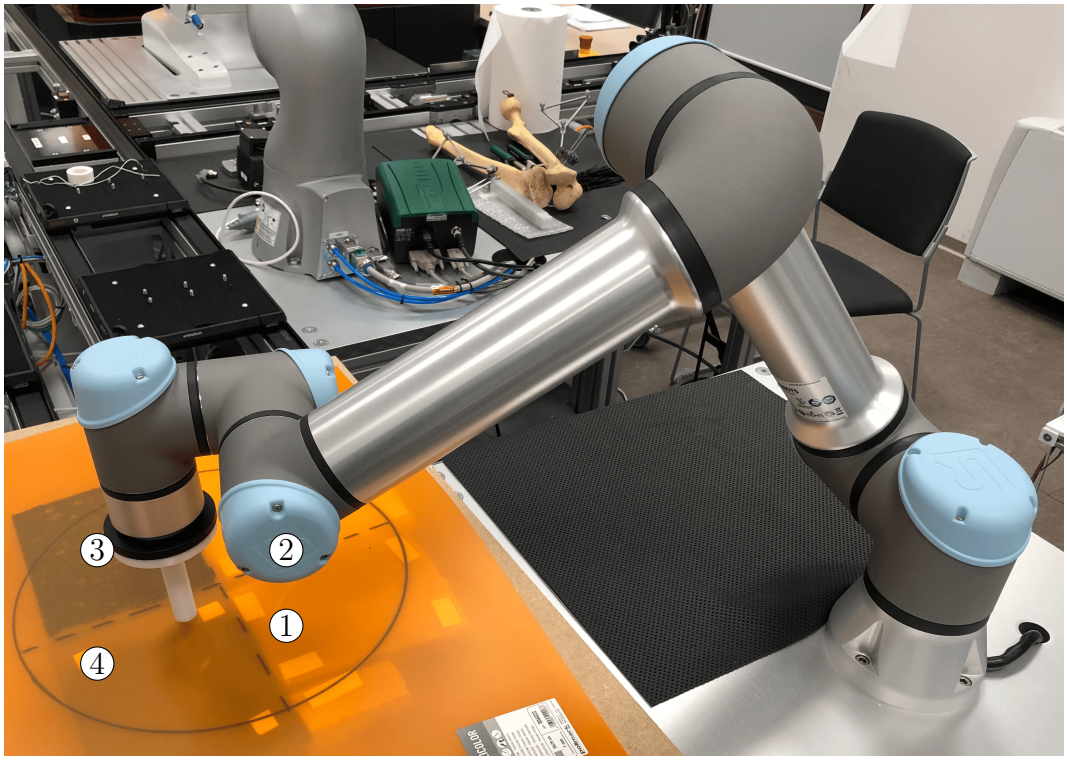


Figure 6.12: The UR5e, equipped with a 3D printed tool, over the environment surface used in the experimental validation. The orange plastic sheet covers the four compliant materials as numbered in the picture.

As usual the applied command to the robot is just the first value

$$\boldsymbol{u}(k) = \hat{\boldsymbol{u}}^{\star}(k). \tag{6.62}$$

Since the environment model $\hat{\boldsymbol{K}}_e$, $\hat{\boldsymbol{B}}_e$, $\hat{\boldsymbol{x}}_e$ is considered time-invariant over the prediction horizon, the contact state $\sigma(t)$ cannot change as well (6.60): in low-motion scenarios, this is not a strong assumption.

6.4.2 Experimental results

The proposed method has been validated on a real setup while executing a polishing-like task on a planar surface with four different kinds of soft environments and different values for $\mathcal{T}^{max}$. The experimental setup consists of a position-controlled 6-DOF Universal Robots UR5e, shown in Figure 6.12, and controlled through the C++ library `ur_rtde`. The interaction forces with the environment are measured using the force-torque sensor installed at the end-effector of the manipulator. The MPC problem is solved using a C++ library developed by the authors (see Appendix A), based on the OSQP solver [172] on a desktop PC running Ubuntu 18.04. The update frequency of the con-

Table 6.1: RMSE and Percentage RMSE of the manipulator and controller estimated torques

	Manipulator (N m, %)	Controller (N m, %)
Joint 1	1.83 - 4.20	8.25 - 18.89
Joint 2	10.12 - 6.97	17.73 - 18.10
Joint 3	6.59 - 9.13	10.55 - 17.45
Joint 4	1.43 - 10.71	2.03 - 16.56
Joint 5	1.21 - 11.40	1.81 - 17.08
Joint 6	0.61 - 6.00	1.54 - 14.61

troller, composed of the MPC and the feedback linearization, is set to 100 Hz (i.e., the sample time is $\Delta t = 10\,\text{ms}$).

The inertia and the viscous friction of the linearised dynamic model (6.42) are

$$\bar{\boldsymbol{M}} = \text{diag}(1.0, 1.0, 1.0, 1.0, 1.0, 0.5)$$
$$\bar{\boldsymbol{D}} = \text{diag}(4.0, 4.0, 4.0, 4.0, 4.0, 2.0)$$

which results in a set of stable second-order linear systems. The prediction and control horizons k_p, k_c of the linear MPC are both set to 15 steps.

Dynamic model identification

The identification process has been performed in free motion with the joints following different sinusoidal trajectories. The estimated parameters by using a different set of joint trajectories. The Root Mean Square Error (RMSE) and the percentage RMSE between the measured and the estimated torques are reported in Table 6.1.

Polishing with interaction force constraints

The polishing-like task is composed of two steps: the first step is to perpendicularly approach and insert the robot's tool into the surface at a specific depth. Once the approach trajectory is completed the manipulator starts following a circular trajectory parallel to the surface at the desired depth to simulate the polishing. During such circular motion, the robot must keep contact with the surface adjusting the tool depth to satisfy the force constraint along the perpendicular axis (in our case the z axis in the robot's base frame).

The surface, shown in Figure 6.12, is a square thin plastic sheet attached to a plywood sheet through a layer composed of different compliant materials arranged as follows: 1), 2) soft materials; 3), 4) stiff materials. The circular

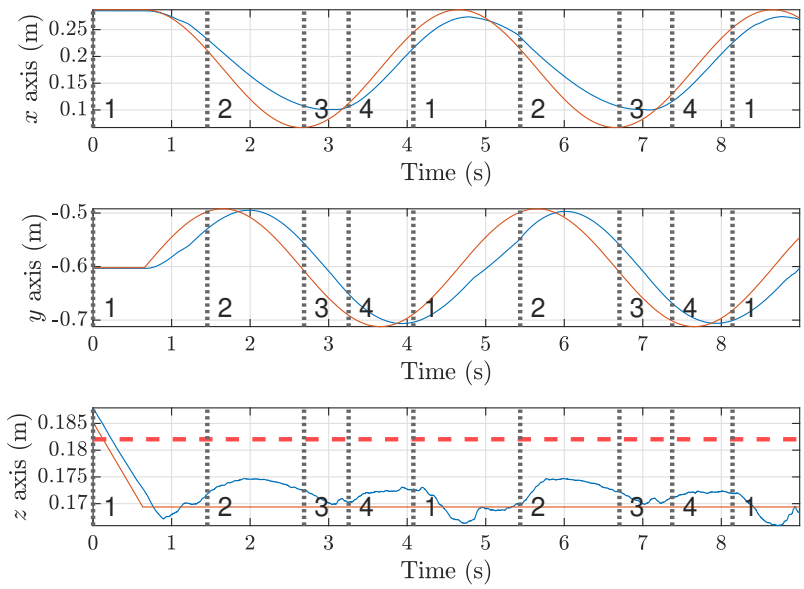
trajectory is designed to have the robot's tool interacts with different quadrant compliant behaviours. When the robot's tip interacts with the environment, $\sigma(t)$ becomes 1, and as described in Section 6.5.2 the model within the MPC is updated to include the estimated parameters of the environment model. Such parameters are refined and updated within the MPC at each control cycle. The circular trajectory chosen for the experiments had a diameter of 22 cm with a travelling time of 4 s for a full round. We validated the proposed methodology using two values for the force upper bounds, 95 N and 75 N such that

$$\text{case 1: } \mathcal{T}^{max} = \begin{bmatrix} 95 & 95 & 95 & 0 & 0 & 0 \end{bmatrix}^T$$
$$\text{case 2: } \mathcal{T}^{max} = \begin{bmatrix} 75 & 75 & 75 & 0 & 0 & 0 \end{bmatrix}^T$$

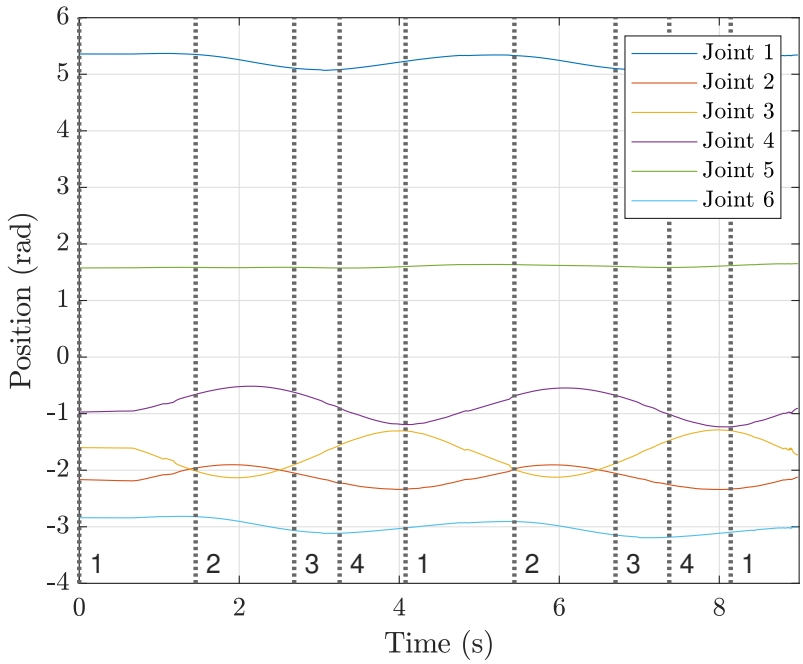
As shown in Figure 6.13a, the robot approaches the desired insertion depth of 12 mm around 1 s and then starts tracking the circular trajectory to simulate the polishing. As shown in Figure 6.14e, the controller is able to provide smooth control action despite the high forces required to perform the task. In both experiments the RMSE of the position tracking error along the x and y axes is of 2 cm. For the z-axis the end-effector reaches different insertion depths based on the different compliance of the materials. In fact, since the force constraint is constant during the experiment the depth is adjusted by the controller to meet when possible the desired position.

Figures 6.14a and 6.14b show the evolution of the environment force during the polishing. The RMSE of the estimation error of the environment force with respect to the force measured by the robot sensor is of 3.6 N in both experiments. Such error, as can be seen, is higher at the beginning of the experiments and when the manipulator interacts with the stiff environments. This is due to the fact that we are not providing any initial guess about environmental compliance ($\boldsymbol{K}_e$). Moreover, the estimation is much more challenging when the stiffness is high since a small variation in the estimated parameter causes a large variation in the estimated force. In any case, the MPC controller correctly limits the estimated interaction force. The unavoidable uncertainty on the estimation can be mitigated by including an offset in the force limit which takes into account such estimation error.

The stiffness coefficients $\boldsymbol{K}_e^i$ of the environment along the z-axis are shown in Figures 6.14c (*case 1*) and 6.14d (*case 2*). The behaviour of the estimation reflects the arrangement of the materials, namely the stiffness is lower for the first and the second layers than for the third and the fourth ones. Even though the parameters slightly differ between the two experiments, their trends are similar and the difference can be due to the estimation of the environment

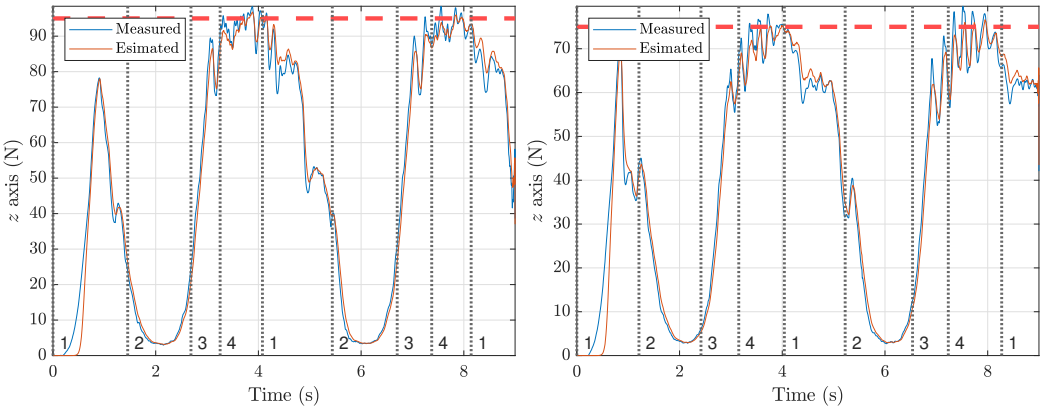


(a) Cartesian space trajectory



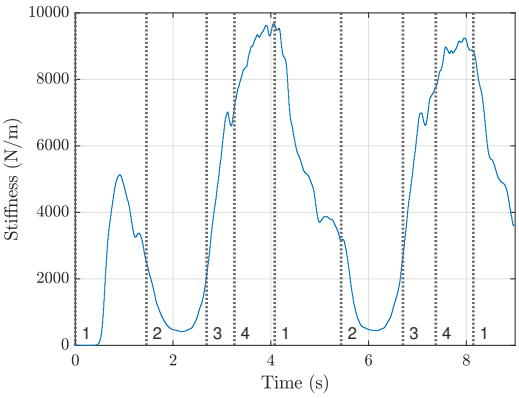
(b) Joint space trajectory

Figure 6.13: Cartesian and joint space robot's trajectories during the 95 N experiment. a) the trajectory and the measured position of the robot's end-effector along the x,y,z axes. The red dashed line is the estimated position of the environment $\boldsymbol{x}_e$, b) the reference joint position $\boldsymbol{q}_r$ computed by the feedback linearization block shown in Figure 6.11.

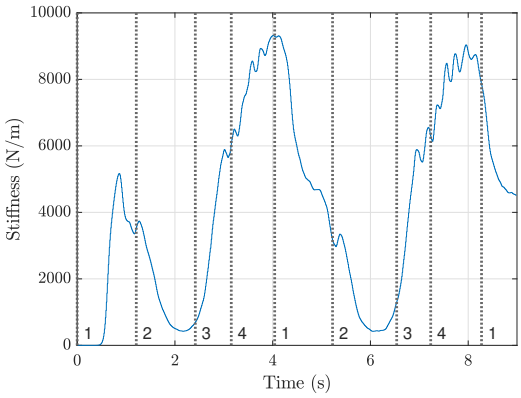


(a) Environment force (95 N)

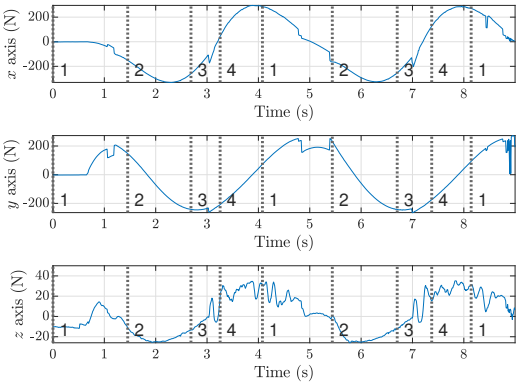
(b) Environment force (75 N)



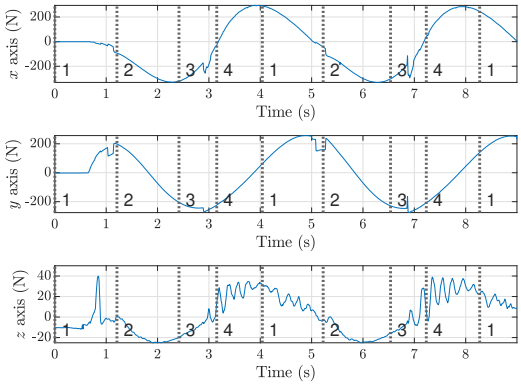
(c) Estimated parameters (95 N)



(d) Estimated parameters (75 N)



(e) Command wrench (95 N)



(f) Command wrench (75 N)

Figure 6.14: Experimental validation for different force limits: 95 N and 75 N; (a),(b) environment forces along the z axis: the blue line is the measured force, the orange line is the estimated force using the RLS, and the dashed red line is the force limit; (c),(d) the estimated environment stiffness along the z axis; (e),(f) MPC optimal command wrench along the x,y,z axes.

position $\boldsymbol{x}_e$ and the accuracy of the force sensor. The estimated vertical position of the environment differs of 0.7 mm between the two experiments. Such a small difference, especially for a high stiffness range causes a difference of several Newton in the estimated force. Nonetheless, the estimated parameters allow to estimate the interaction forces quite accurately.

In Figures 6.14a and 6.14b, due to the estimation inaccuracy the measured interaction force is slightly higher but anyway within the 3% and 6% of the threshold $\mathcal{T}^{max}$ for the 95 N and 75 N, respectively. Figure 6.14e and Figure 6.14f show the optimal wrench commands computed by the MPC. On all Cartesian axes, the control action is smooth and is not affected by the transitions between the different materials. These commands are then converted into the desired joint positions by the feedback linearization control loop (6.48) as shown in Figure 6.13b.

6.5 Passivity-based adaptive linear MPC

In the previous section we show how we can control a robotic manipulator using an adaptive linear MPC and enforcing interaction force constraints in real-time. In this Section we will now extend the previous unilateral approach to model the whole teleoperation system as an adaptive linear P-MPC. As in the non-linear case the teleoperation system is composed of two manipulators, one interacting with the operator and the other interacting with the environment. The two manipulators are fully actuated, gravity compensated and modelled by the following n-DOFs Euler-Lagrangian system

$$\boldsymbol{M}(\boldsymbol{x})\ddot{\boldsymbol{x}} + \boldsymbol{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})\dot{\boldsymbol{x}} + \boldsymbol{D}(\boldsymbol{x})\dot{\boldsymbol{x}} = \boldsymbol{u} + \boldsymbol{h} \tag{6.63}$$

where $\boldsymbol{x}$ represents the Cartesian position and orientation, $\dot{\boldsymbol{x}}$ represents the Cartesian linear and angular velocities, $\boldsymbol{u}$ and $\boldsymbol{h}$ are the control input and the external torque. Let $H(t)$ be the Hamiltonian function representing the stored energy of the system (6.63)

$$H(t) = -\frac{1}{2}\dot{\boldsymbol{x}}^T\boldsymbol{M}(\boldsymbol{x})\dot{\boldsymbol{x}} + \dot{\boldsymbol{x}}^T(\boldsymbol{u} + \boldsymbol{h}). \tag{6.64}$$

The passivity of the system follows from the power balance [14]

$$\dot{H}(t) \leq \dot{\boldsymbol{x}}^T(\boldsymbol{u} + \boldsymbol{h}). \tag{6.65}$$

In order to proceed with the linear approximation of the tank dynamic we redefine the policy for the energy transfer between the energy tanks as follows

$$\begin{cases} P_o^{out}(t) & = (1-\sigma_o)(\dot{\boldsymbol{x}}_o^T\boldsymbol{D}_o\dot{\boldsymbol{x}}_o + x_{t_o}u_{t_o}) + E_r^{req}(t-\delta)\beta_o x_{t_o}\bar{p} \\ P_r^{out}(t) & = (1-\sigma_r)(\dot{\boldsymbol{x}}_r^T\boldsymbol{D}_r\dot{\boldsymbol{x}}_r + x_{t_r}u_{t_r}) + E_o^{req}(t-\delta)\beta_r x_{t_r}\bar{p} \end{cases} \tag{6.66}$$

where $\bar{p} \in [0, 1]$ is a design parameter representing the percentage of the available energy in the tank which flows from one tank to the other. The power dissipated by the current desired command at time t, $P^u(t)$, is defined as

$$P^u(t) = u_t(t)x_t(t). \tag{6.67}$$

Lemma 2. $P^{out}(t) \geq 0$

Proof. Since $\sigma, E^{req}, \beta \in \{0, 1\}$ and $\bar{p} \geq 0$ we get

$$E^{req}(t - \delta)\beta\bar{p} \geq 0 \tag{6.68}$$

Thus, from (6.66) it follows that $P^{out}(t)$ is positive if and only if

$$(1 - \sigma)(\dot{\boldsymbol{x}}^T \boldsymbol{D}\dot{\boldsymbol{x}} + x_t u_t) \geq -E^{req}(t - \delta)\beta\bar{p} \tag{6.69}$$

If $\sigma = 1$, (6.69) becomes

$$-E^{req}(t - \delta)\beta\bar{p} \leq 0 \tag{6.70}$$

that is always true thanks to (6.68). If $\sigma = 0$, from (5.12) we have

$$\dot{\boldsymbol{x}}^T \boldsymbol{D}\dot{\boldsymbol{x}} + x_t u_t \geq 0 \tag{6.71}$$

which satisfy (6.69). This is the only case where the input term contributes to $P^{out}(t)$. As a consequence, $P^{out} \geq 0$ and this proves the lemma. □

Proposition 3. *The definition of $P^{out}(t)$ in (6.66) preserves the passivity with respect to the pair $(\boldsymbol{\tau}_o, \dot{\boldsymbol{x}}_o)$ and $(\boldsymbol{\tau}_r, \dot{\boldsymbol{x}}_r)$ even if the system is affected by communication delay.*

Proof. Let the total energy stored $\mathcal{H}$ in the system be

$$\mathcal{H} = H_o(t) + H_r(t) + E_o(t) + E_r(t) + H^{ch}(t) \tag{6.72}$$

where $H_o(t)$ and $H_r(t)$ are the energy associated to (6.63), $E_o(t)$ and $E_r(t)$ are the energy stored in the tanks and $H^{ch}(t)$ is the energy flowing in the communication channel. Considering the derivative $\dot{\mathcal{H}}$ of the Hamiltonian $\mathcal{H}$

$$\dot{\mathcal{H}} = \dot{H}_o(t) + \dot{H}_r(t) + \dot{E}_o(t) + \dot{E}_r(t) + \dot{H}^{ch}(t) \tag{6.73}$$

and using (5.27) we end up with

$$\begin{aligned}\dot{\mathcal{H}} = &-\dot{\boldsymbol{x}}^T \boldsymbol{D}_o\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}^T \boldsymbol{D}_r\dot{\boldsymbol{x}} + \boldsymbol{u}_o^T\dot{\boldsymbol{x}}_o + \boldsymbol{u}_r^T\dot{\boldsymbol{x}}_r + \dot{\boldsymbol{x}}^T \boldsymbol{D}_o\dot{\boldsymbol{x}} + \sigma_o P_o^{in}(t) - P_o^{out}(t) + \\ &+ P_o^u(t) + \dot{\boldsymbol{x}}^T \boldsymbol{D}_r\dot{\boldsymbol{x}} + \sigma_r P_r^{in}(t) - P_r^{out}(t) + P_r^u(t) + \boldsymbol{\tau}_o^T\dot{\boldsymbol{x}}_o + \boldsymbol{\tau}_r^T\dot{\boldsymbol{x}}_r + \dot{H}^{ch}(t)\end{aligned} \tag{6.74}$$

where $P_o^u(t) = -\boldsymbol{u}_o^T \dot{\boldsymbol{x}}_o$ and $P_r^u(t) = -\boldsymbol{u}_r^T \dot{\boldsymbol{x}}_r$ are the powers needed to execute the current robot commands. As shown in [119] the power flowing through the communication channel can be written as

$$\dot{H}^{ch}(t) = P_o^{out}(t) + P_r^{out}(t) - P_o^{out}(t-\delta) - P_r^{out}(t-\delta) \tag{6.75}$$

where $\delta \geq 0$ is the communication delay. We can rewrite (6.74) using (6.66) and (6.75) as

$$\begin{aligned}\dot{\mathcal{H}} = &-(1-\sigma_o)D_o(t) - (1-\sigma_r)D_r(t) - (1-\sigma_r)P_o^{out}(t-\delta) - \\ &+ (1-\sigma_o)P_r^{out}(t-\delta) + \boldsymbol{\tau}_o \dot{\boldsymbol{x}}_o + \boldsymbol{\tau}_r \dot{\boldsymbol{x}}_r.\end{aligned} \tag{6.76}$$

Since $\sigma_o, \sigma_r, \in \{0, 1\}$ we can upper bound $\dot{\mathcal{H}}$ by exploiting the inequalities $u^T \dot{\boldsymbol{x}} \leq u^T \dot{\boldsymbol{x}} + \sigma P^{out}(t)$ and $P^{out}(t) \geq 0$

$$\dot{\mathcal{H}} \leq \boldsymbol{\tau}_o^T \dot{\boldsymbol{x}}_o + \boldsymbol{\tau}_r^T \dot{\boldsymbol{x}}_r \tag{6.77}$$

which proves the theorem. □

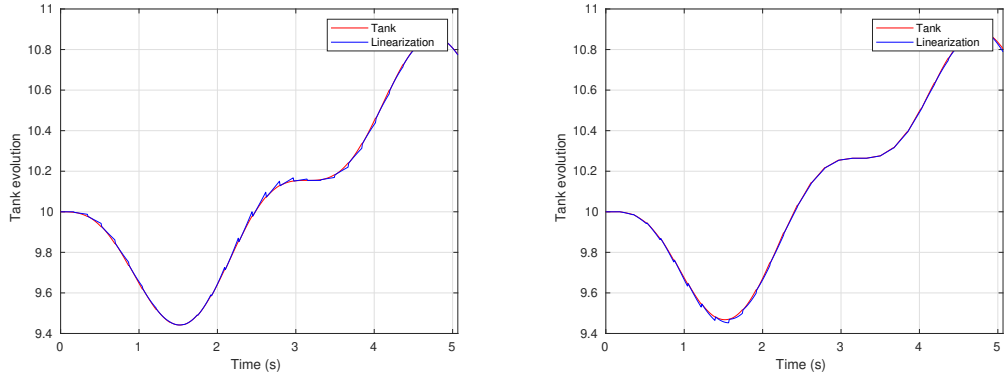
6.5.1 Linear approximation of the tank dynamics

The energy-storing system defined in (5.27) is non-linear since the energy must be computed at the power port of the tank and this prevents the definition of an efficient quadratic optimization problem (QP). For this reason we propose a first-order linear under-approximation of the tank dynamic $\dot{\hat{x}}_t$ around the actual operating point $(\hat{\bar{x}}_t, \dot{\bar{\boldsymbol{x}}}, \bar{\boldsymbol{u}})$. The approximated tank dynamics can be guaranteed to be always bounded by (5.27) under a set of linear constraints on the power port u_t. This condition is necessary since as shown in Figure 6.15 the linear approximation by itself can cause an over-estimation of the available energy causing possible instability. The coordinates are chosen to derive the under-approximation conditions on the manipulated variables $\boldsymbol{u}$ and on the robot's state $\dot{\boldsymbol{x}}$. We initially derived such conditions for an energy tank model and then we prove that such conditions hold also in case of interconnected tank dynamics. The approximated dynamics is meant to be used just within the prediction horizon of the MPC controller. It's value is reinitialized at each control cycle with the real energy value stored in the non-linear tank dynamic outside the MPC.

Single energy tank

Let $\dot{x}_t$ be the energy tank state that takes into account (6.63)

$$\dot{x}_t = \sigma \left(\frac{\dot{\boldsymbol{x}}^T \boldsymbol{D} \dot{\boldsymbol{x}}}{x_t} + u_t(t) \right) \tag{6.78}$$



(a) Linearization (b) Constrained linearization

Figure 6.15: Comparison between the non-linear tank dynamic (in red) and the tank linearization (in blue). a) the naive linearization, b) the linearization under the proposed constraints.

where σ in this case is used to disconnect the tank from the manipulator to limit the energy stored in it. The power interconnection with the manipulator is done by

$$u_t(t) = -\boldsymbol{u}^T(t)\dot{\boldsymbol{x}}(t). \tag{6.79}$$

To get rid of the non-linear condition in σ, the linearization is performed only in the case of $\sigma = 1$ (namely when the tank is coupled with the robot). Under the aforementioned consideration the linearization of (6.78) can be defined as

$$\begin{aligned}\dot{\hat{x}}_t = &\frac{\dot{\bar{\boldsymbol{x}}}^T \boldsymbol{D} \dot{\bar{\boldsymbol{x}}}}{\hat{\bar{x}}_t} - \bar{\boldsymbol{u}}^T \dot{\bar{\boldsymbol{x}}} - \frac{\dot{\bar{\boldsymbol{x}}}^T \boldsymbol{D} \dot{\bar{\boldsymbol{x}}}}{\hat{\bar{x}}_t^2}(\hat{x}_t(t) - \hat{\bar{x}}_t) - \dot{\bar{\boldsymbol{x}}}^T(\boldsymbol{u}(t) - \bar{\boldsymbol{u}}) + \\ &+ (\frac{2\boldsymbol{D}\dot{\bar{\boldsymbol{x}}}}{\hat{\bar{x}}_t} - \bar{\boldsymbol{u}})^T(\dot{\boldsymbol{x}} - \dot{\bar{\boldsymbol{x}}}).\end{aligned} \tag{6.80}$$

To guarantee the solutions of $\dot{\hat{x}}_t$ to be an under-approximation of the solutions of $\dot{x}_t$ in (6.78) we need to satisfy $\forall t_0 \in \mathbb{R}^+$

$$x_t(t_0) + \int_{t_0}^{t_0+T} \dot{\hat{x}}_t dt \leq x_t(t_0) + \int_{t_0}^{t_0+T} \dot{x}_t dt \tag{6.81}$$

where T is the prediction horizon of the MPC. The inequality (6.81) is equivalent to

$$\int_{t_0}^{t_0+T} (\dot{\hat{x}}_t - \dot{x}_t) dt \leq 0 \tag{6.82}$$

and is satisfied if

$$\dot{\hat{x}}_t - \dot{x}_t \leq 0. \tag{6.83}$$

Substituting (6.78) in (6.83)

$$\dot{\hat{x}}_t \leq \sigma \left(\frac{\dot{\boldsymbol{x}}^T \boldsymbol{D} \dot{\boldsymbol{x}}}{x_t} + u_t(t) \right) \tag{6.84}$$

and considering the fact that $\dot{\boldsymbol{x}}^T \boldsymbol{D} \dot{\boldsymbol{x}} \geq 0$, the condition (6.83) holds if and only if

$$\dot{\hat{x}}_t \leq -\sigma \boldsymbol{u}(t)^T \dot{\boldsymbol{x}}. \tag{6.85}$$

As before, not considering the damping term in (6.80), $\boldsymbol{D} = 0$

$$-\dot{\bar{\boldsymbol{x}}}^T \boldsymbol{u}(t) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}} + \bar{\boldsymbol{u}}^T \dot{\bar{\boldsymbol{x}}} \leq \dot{\hat{x}}_t \leq -\sigma \boldsymbol{u}(t)^T \dot{\boldsymbol{x}} \tag{6.86}$$

we can satisfy (6.86) if and only if

$$-\dot{\bar{\boldsymbol{x}}}^T \boldsymbol{u}(t) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}} + \bar{\boldsymbol{u}}^T \dot{\bar{\boldsymbol{x}}} \leq -\sigma \boldsymbol{u}(t)^T \dot{\boldsymbol{x}}. \tag{6.87}$$

Is it worth mentioning that such new condition does not depends on the tank level $\hat{x}_t$. In case the real tank is disconnected from the manipulator (6.63), i.e., $\sigma = 0$, the set

$$\mathcal{P}_0 = \{(\dot{\boldsymbol{x}}, \boldsymbol{u}) | (\bar{\boldsymbol{u}} - \boldsymbol{u}(t))^T \dot{\bar{\boldsymbol{x}}} \leq \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}}\} \tag{6.88}$$

defines conditions on $(\dot{\boldsymbol{x}}, \boldsymbol{u})$ to be solutions to (6.87) with $\sigma = 0$. On the other hand, when the real tank is coupled with the manipulator, i.e., $\sigma = 1$, we can rewrite (6.87) as

$$(\boldsymbol{u}(t) - \bar{\boldsymbol{u}})^T (\dot{\boldsymbol{x}} - \dot{\bar{\boldsymbol{x}}}) \leq 0 \tag{6.89}$$

where the set

$$\begin{aligned} \mathcal{P}_1 = \{(\dot{\boldsymbol{x}}, \boldsymbol{u}) | (\boldsymbol{u}_i(t) \leq \bar{\boldsymbol{u}}_i \wedge \dot{\boldsymbol{x}}_i \geq \dot{\bar{\boldsymbol{x}}}_i) \vee \\ \vee (\boldsymbol{u}_i(t) \geq \bar{\boldsymbol{u}}_i \wedge \dot{\boldsymbol{x}}_i \leq \dot{\bar{\boldsymbol{x}}}_i), \forall i = 1 \ldots n\} \end{aligned} \tag{6.90}$$

defines the sufficient conditions on $(\dot{\boldsymbol{x}}, \boldsymbol{u})$ to be solutions of (6.87) with $\sigma = 1$. The conditions in the set $\mathcal{P}_1$ are intended to be applied component-wise.

Interconnected energy tanks

Let now consider the case in which two energy tanks are connected together by means of the power signals P^{in} and P^{out}. This scenario, as we saw in Chapter 5, is particularly useful in case of bilateral teleoperation since allows to share energy between the two teleoperation's sides. As in the single tank case we need to find a set of constraints such that the under-approximation condition (6.83) holds. So, let $\dot{x}_t$ be the interconnected tank dynamics defined in (5.27) such that substituting (5.27) in (6.83) we get

$$\dot{\hat{x}}_t \leq \frac{\dot{\boldsymbol{x}}^T \boldsymbol{D} \dot{\boldsymbol{x}} + \sigma P^{in} - P^{out}(t)}{x_t} + u_t(t). \tag{6.91}$$

Since $\dot{\boldsymbol{x}}^T \boldsymbol{D} \dot{\boldsymbol{x}} + \sigma P^{in} \geq 0$ by (6.63), (5.29), the under-approximation condition (6.91) is satisfied if and only if

$$\dot{\hat{x}}_t \leq \frac{-P^{out}(t)}{x_t} + u_t(t) \tag{6.92}$$

and substituting (6.66) in (6.92) the inequality is satisfied if and only if

$$\dot{\hat{x}}_t \leq -(1-\sigma)u_t + E_r^{req}(t-\delta)\beta_o\bar{p} + u_t(t). \tag{6.93}$$

Since $E^{req}, \beta \in \{0,1\}$, we can derive the bound conditions for the under-approximation in the worse case scenario. In such scenario, the tank leaks a fixed amount of energy $\bar{p}$ over time. With this further consideration the inequality (6.93) reduces to

$$\dot{\hat{x}}_t \leq \underbrace{-(1-\sigma)u_t(t) + u_t(t) + \bar{p}}_{\dot{x}_t}. \tag{6.94}$$

Let now define the linearization for the interconnected tanks $\dot{\hat{x}}_t$ as follows

$$\dot{\hat{x}}_t = -\dot{\bar{\boldsymbol{x}}}^T\boldsymbol{u}(t) - \bar{\boldsymbol{u}}^T\dot{\boldsymbol{x}}(t) + \dot{\bar{\boldsymbol{x}}}^T\bar{\boldsymbol{u}} + \bar{p}. \tag{6.95}$$

Also in this case the linearization is performed to keep the tank and the manipulator always coupled during the optimization, i.e., with $\sigma = 1$. Substituting (6.79) in (6.94) we get the under-approximation condition for the interconnected energy tanks

$$-\dot{\bar{\boldsymbol{x}}}^T\boldsymbol{u}(t) - \bar{\boldsymbol{u}}^T\dot{\boldsymbol{x}} + \dot{\bar{\boldsymbol{x}}}^T\bar{\boldsymbol{u}} \leq (1-\sigma)\boldsymbol{u}^T(t)\dot{\boldsymbol{x}} - \boldsymbol{u}^T(t)\dot{\boldsymbol{x}}. \tag{6.96}$$

which, thanks to (6.96), can be rewritten as

$$-\dot{\bar{\boldsymbol{x}}}^T\boldsymbol{u}(t) - \bar{\boldsymbol{u}}^T\dot{\boldsymbol{x}} + \dot{\bar{\boldsymbol{x}}}^T\bar{\boldsymbol{u}} \leq -\sigma\boldsymbol{u}^T(t)\dot{\boldsymbol{x}}. \tag{6.97}$$

The conditions for the interconnected case are guaranteed by the same passive sets $\mathcal{P}_1$ and $\mathcal{P}_0$.

Stability analysis

The proposed linearized tank dynamics can be used to enforce the closed loop stability of the following continuous-time MPC problem $\mathbb{P}(\boldsymbol{z}, \hat{x}_t)$

$$\begin{aligned}
V(\boldsymbol{z}, \hat{x}_t) = \quad & \min_{\boldsymbol{u}(\cdot)} \quad \int_0^T \mathcal{C}(\boldsymbol{z}(\tau), \boldsymbol{u}(\tau))d\tau \\
& \text{subject to} \quad \dot{\boldsymbol{z}} = f(\boldsymbol{z}, \boldsymbol{u}) \\
& \qquad \dot{\hat{x}}_t = -\dot{\bar{\boldsymbol{x}}}^T\boldsymbol{u}(t) - \bar{\boldsymbol{u}}^T\dot{\boldsymbol{x}}(t) + \dot{\bar{\boldsymbol{x}}}^T\bar{\boldsymbol{u}} + \bar{p} \\
& \qquad \boldsymbol{z}(0) = \boldsymbol{z}_0,\ \hat{x}_t(0) = x_{t_0} \\
& \qquad \boldsymbol{z} \in \mathcal{Z},\ \boldsymbol{u} \in \mathcal{U} \\
& \qquad \boldsymbol{z}(T) = 0 && (6.98) \\
& \qquad \hat{x}_t \geq \varepsilon && (6.99) \\
& \qquad (\boldsymbol{u}, \dot{\boldsymbol{x}}) \in \mathcal{P}_\sigma && (6.100)
\end{aligned}$$

where $V(\boldsymbol{z}, \hat{x}_t)$ is the value function, $\sigma \in \{0, 1\}$ and $f(\boldsymbol{z}, \boldsymbol{u})$ is the system controlled by the input

$$\boldsymbol{u} = \kappa(\tau; \boldsymbol{z}, x_t) = \boldsymbol{u}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t), \quad \tau \in [0, \Delta t] \tag{6.101}$$

with $\boldsymbol{z} = \begin{bmatrix} \boldsymbol{x}^T & \dot{\boldsymbol{x}}^T \end{bmatrix}^T$ and where $\boldsymbol{u}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t)$ denotes the optimal control input at time t of $\mathbb{P}(\boldsymbol{z}, \hat{x}_t)$. The tank level variable σ is initialized at each control cycle and kept constant along the prediction horizon T. The cost $\mathcal{C}(\cdot)$ is a positive definite function and it satisfies $\mathcal{C}(0, 0) = 0$. The sets $\mathcal{Z}$ and $\mathcal{U}$ are compact and represent the state and input constraints, respectively. Similar to [173], [174], we are keeping track of the energy accumulated in the system to enforce the closed-loop passivity by (6.99). In our case, we don't compute the non-linear storage function x_t within the MPC but a linear under-approximation $\hat{x}_t$ by enforcing the additional constraint (6.100).

Proposition 4. *The aforementioned MPC controller, augmented with the linear under-approximation $\hat{x}_t$ and constrained with* (6.99) *and* (6.100) *guarantees the open and closed loop stability.*

Proof. Let $(\boldsymbol{z}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t), \boldsymbol{u}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t), \hat{x}_t^\star(\tau; \boldsymbol{z}, \hat{x}_t), t)$, $\tau \in [0, T]$ be the optimal solution at time t of $\mathbb{P}(\boldsymbol{z}, \hat{x}_t)$. Now we prove that at time $t + \Delta t$ the truncated input function

$$\tilde{\boldsymbol{u}}(\tau) = \begin{cases} \boldsymbol{u}^\star(\tau + \Delta t; \boldsymbol{z}, x_t, t) & \tau \in [0, T - \Delta t] \\ 0 & \tau \in [T - \Delta t, T] \end{cases} \tag{6.102}$$

is a feasible solution to $\mathbb{P}(\boldsymbol{z}^\star(\Delta t, \boldsymbol{z}, \hat{x}_t, t), \hat{x}_t^\star(\Delta t, \boldsymbol{z}, \hat{x}_t, t))$. Let

$$\tilde{\boldsymbol{z}}(\tau) = \begin{cases} \boldsymbol{z}^\star(\tau + \Delta t; \boldsymbol{z}, x_t, t) & \tau \in [0, T - \Delta t] \\ 0 & \tau \in [T - \Delta t, T] \end{cases} \tag{6.103}$$

be the truncated state function. By construction $\tilde{\boldsymbol{u}}(\tau)$ is within the sets $\mathcal{U}$ and $\mathcal{P}$ and $\tilde{\boldsymbol{z}}(\tau)$ is contained in $\mathcal{Z}$ and satisfies the terminal condition (6.98). Let at time t the passivity condition (6.99) satisfied such that

$$\hat{x}_t(T) = \hat{x}_t(0) + \int_0^T (-\dot{\bar{\boldsymbol{x}}}^T \boldsymbol{u}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) + \dot{\bar{\boldsymbol{x}}}^T \bar{\boldsymbol{u}} + \bar{p}) d\tau \geq \varepsilon \tag{6.104}$$

where $\hat{x}_t(0)$ is the initial condition at time t. At time $t + \Delta t$ the evolution of

$\hat{x}_t(\cdot)$ when the previous feasible solution at time t is employed is

$$\begin{aligned}\hat{\tilde{x}}_t(T) &= \hat{\tilde{x}}_t(0) + \int_0^T (-\dot{\tilde{\boldsymbol{x}}}^T \tilde{\boldsymbol{u}}(\tau) - \bar{\boldsymbol{u}}^T \dot{\tilde{\boldsymbol{x}}}(\tau) + \dot{\tilde{\boldsymbol{x}}}^T \bar{\boldsymbol{u}} + \bar{p}) d\tau \\ &= \hat{\tilde{x}}_t(0) + \int_{\Delta t}^T (-\dot{\tilde{\boldsymbol{x}}}^T \boldsymbol{u}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) + \dot{\tilde{\boldsymbol{x}}}^T \bar{\boldsymbol{u}} + \bar{p}) d\tau\end{aligned} \tag{6.105}$$

where $\hat{\tilde{x}}_t(0) = \hat{x}_t(\Delta t)$. Adding and subtracting the term

$$\int_0^{\Delta t} (-\dot{\tilde{\boldsymbol{x}}}^T \boldsymbol{u}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) + \dot{\tilde{\boldsymbol{x}}}^T \bar{\boldsymbol{u}} + \bar{p}) d\tau \tag{6.106}$$

in (6.105), we get

$$\begin{aligned}\hat{\tilde{x}}_t(T) &= \hat{\tilde{x}}_t(0) + \int_0^T (-\dot{\tilde{\boldsymbol{x}}}^T \boldsymbol{u}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) + \dot{\tilde{\boldsymbol{x}}}^T \bar{\boldsymbol{u}} + \bar{p}) d\tau \\ &\quad - \int_0^{\Delta t} (-\dot{\tilde{\boldsymbol{x}}}^T \boldsymbol{u}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) + \dot{\tilde{\boldsymbol{x}}}^T \bar{\boldsymbol{u}} + \bar{p}) d\tau\end{aligned} \tag{6.107}$$

which can be rewritten as

$$\hat{\tilde{x}}_t(T) = \hat{x}_t(\Delta t) + \int_{\Delta t}^T (-\dot{\tilde{\boldsymbol{x}}}^T \boldsymbol{u}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}}^\star(\tau; \boldsymbol{z}, \hat{x}_t, t) + \dot{\tilde{\boldsymbol{x}}}^T \bar{\boldsymbol{u}} + \bar{p}) d\tau. \tag{6.108}$$

By the definition of the approximated tank dynamics (6.95), we can rewrite (6.108) such that

$$\hat{\tilde{x}}_t(T) = \hat{x}_t(T) \geq \varepsilon \tag{6.109}$$

which shows the feasibility of the integral constraint (6.99) which enforces the passivity of the open-loop controller. Moreover, the feasibility at any time guarantees also the invariance of the set

$$\mathcal{X} = \{\boldsymbol{z} \in \mathcal{Z} | \mathbb{P}(\boldsymbol{z}, T^{ini}) \text{ is feasible}\} \tag{6.110}$$

which contains all the initial states $\boldsymbol{z}$ such that a solution to $\mathbb{P}(\boldsymbol{z}, T^{ini})$ exists, where $T^{ini} > \varepsilon$ is the initial energy level stored in the tank.

Let now $\mathcal{X}_{\hat{x}_{t_0}} = \{\boldsymbol{z} \in \mathcal{Z} | \mathbb{P}(\boldsymbol{z}, \hat{x}_{t_0}) \text{ is feasible}\}$ be the set of feasible states $\boldsymbol{z}$ with $\hat{x}_{t_0}$ be the amount of energy stored in the tank at time t. To guarantee the closed-loop stability $\forall \boldsymbol{z}(0) \in \mathcal{X}$ with $\hat{x}_t(0) = T^{ini}$ we need to satisfy for all $t \geq 0$ and $\boldsymbol{z} \in \mathcal{X}_{\hat{x}_{t_0}}$

$$\int_0^t (-\dot{\tilde{\boldsymbol{x}}}^T \kappa(\tau; \boldsymbol{z}, x_t) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}}(\tau) + \dot{\tilde{\boldsymbol{x}}}^T \bar{\boldsymbol{u}} + \bar{p}) d\tau \geq \varepsilon. \tag{6.111}$$

Thanks to the invariance set (6.110) the proposed MPC scheme is always feasible if it is feasible at time t_0, and $\forall \boldsymbol{z} \in \mathcal{X}_{\hat{x}_{t_0}}$ the following condition holds

$$\hat{x}_t(t_0) + \int_{t_0}^{t_0+\Delta t} (-\dot{\bar{\boldsymbol{x}}}^T \kappa(\tau; \boldsymbol{z}, x_t) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}}(\tau) + \dot{\bar{\boldsymbol{x}}}^T \bar{\boldsymbol{u}} + \bar{p}) d\tau + \\ + \int_{t_0+\Delta t}^{t_0+T} (-\dot{\bar{\boldsymbol{x}}}^T \boldsymbol{u}(\tau) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}}(\tau) + \dot{\bar{\boldsymbol{x}}}^T \bar{\boldsymbol{u}} + \bar{p}) d\tau \geq \varepsilon \tag{6.112}$$

where

$$\hat{x}_t(t_0) = \int_0^{t_0} (-\dot{\bar{\boldsymbol{x}}}^T \kappa(\tau; \boldsymbol{z}, x_t) - \bar{\boldsymbol{u}}^T \dot{\boldsymbol{x}}(\tau) + \dot{\bar{\boldsymbol{x}}}^T \bar{\boldsymbol{u}} + \bar{p}) d\tau \tag{6.113}$$

and the contribution from $t_0 + \Delta t$ to $t_0 + T$ in (6.112) is given by the open-loop controller. Since (6.112) holds for any $t_0 > 0$ the closed-loop controller (6.101) is passive $\forall \boldsymbol{z} \in \mathcal{X}_{\hat{x}_{t_0}}$. □

6.5.2 Adaptive linear MPC modelling

Since the teleoperation control architecture is distributed between operator and remote sides, the whole control system has been replicated at both sides of the communication channel, as shown in Figure 6.16. The adaptive MPC plant is composed of two linearized manipulator dynamics, the linear approximation of the energy tanks, and the environment model. Such model is not constant over time: the local linearization of the tank and the switching behaviour (free-motion vs contact) of the remote side system should be recomputed at each control cycle. To mitigate the effect of the linear under-approximation the tanks inside the MPC are reinitialized at each cycle with the energy level contained in the real tanks outside the MPC. Moreover, the whole system is subjected to different exogenous inputs depending at which side of the teleoperation the controller is.

Let $\boldsymbol{z} \in \mathbb{R}^N$ be the state vector of the MPC where $N = 4n + 1$ and let the dynamic model within the MPC system be

$$\begin{cases} \dot{\boldsymbol{z}}(t) = \boldsymbol{A}\boldsymbol{z}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{B}^d \boldsymbol{d}(t) \\ \boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{z}(t) + \boldsymbol{D}^d \boldsymbol{d}(t) \end{cases} \tag{6.114}$$

where $\boldsymbol{d}(t) \in \mathbb{R}^{3n+1}$ is the measured exogenous input of the system and depends on the controlled manipulator. At the operator side is equal to

$$\boldsymbol{d}_o = \begin{bmatrix} \boldsymbol{h}_o \\ \boldsymbol{u}_r \\ \psi(t)\boldsymbol{x}_e \\ (\dot{\bar{\boldsymbol{x}}}_o^T \bar{\boldsymbol{u}}_o + \bar{p}_o) \end{bmatrix} \tag{6.115}$$

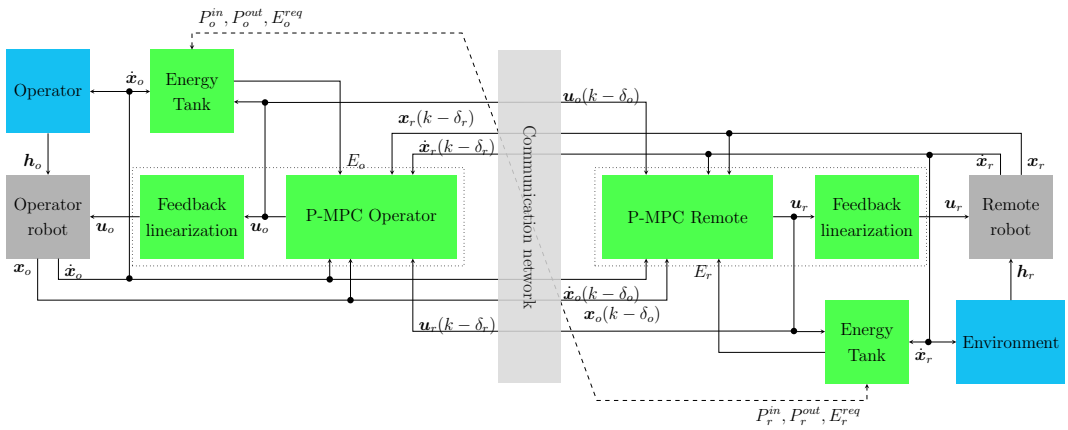


Figure 6.16: The MPC-based control architecture. The green blocks are the control modules, the blue blocks are the external torque sources, the grey blocks are the manipulators and the light grey is the communication channel.

while at the remote side is defined as

$$\boldsymbol{d}_r = \begin{bmatrix} \boldsymbol{h}_o \\ \boldsymbol{u}_o \\ \psi(t)\boldsymbol{x}_e \\ (\dot{\bar{\boldsymbol{x}}}_r^T \bar{\boldsymbol{u}}_r + \bar{p}_r) \end{bmatrix}. \tag{6.116}$$

The state evolution matrix $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ is time-varying and is composed of the state evolution matrices of the both manipulators and the tank of the controlled teleoperation side. The full state evolution matrix

$$\boldsymbol{A} = \begin{cases} \boldsymbol{A}_o, & \text{MPC at the operator side} \\ \boldsymbol{A}_r, & \text{MPC at the remote side} \end{cases}$$

within the MPC at the operator side is

$$\boldsymbol{A}_o = \begin{pmatrix} \boldsymbol{A}_{1,1} & 0^{2n\times 1} & 0^{2n\times 2n} \\ 0^{1\times n} - \bar{\boldsymbol{u}}_o^T & 0^{1\times 1} & 0^{1\times 2n} \\ 0^{2n\times 2n} & 0^{2n\times 1} & \boldsymbol{A}_{2,2} \end{pmatrix} \tag{6.117}$$

while at the remote side is

$$\boldsymbol{A}_r = \begin{pmatrix} \boldsymbol{A}_{1,1} & 0^{2n\times 1} & 0^{2n\times 2n} \\ 0^{1\times 2n} & 0^{1\times 1} & 0^{1\times n} - \bar{\boldsymbol{u}}_r^T \\ 0^{2n\times 2n} & 0^{2n\times 1} & \boldsymbol{A}_{2,2} \end{pmatrix} \tag{6.118}$$

where $\boldsymbol{A}_{1,1}$ and $\boldsymbol{A}_{2,2}$ are the operator and remote side manipulators state evolution matrices, respectively. The operator side robot model is

$$\boldsymbol{A}_{1,1} = \begin{pmatrix} 0^{n\times n} & \boldsymbol{I}^{n\times n} \\ 0^{n\times n} & -\boldsymbol{M}_o^{-1}\boldsymbol{D}_o \end{pmatrix} \tag{6.119}$$

while the remote side robot model $\boldsymbol{A}_{2,2}$ is time varying since the manipulator switches over time between the free-motion and contact state. In the free-motion case (i.e., $\psi(t) = 0$)

$$\boldsymbol{A}_{2,2} = \begin{pmatrix} 0^{n\times n} & \boldsymbol{I}^{n\times n} \\ 0^{n\times n} & -\boldsymbol{M}_r^{-1}\boldsymbol{D}_r \end{pmatrix} \tag{6.120}$$

while in the contact case (i.e. $\psi(t) = 1$)

$$\boldsymbol{A}_{2,2} = \begin{pmatrix} 0^{n\times n} & \boldsymbol{I}^{n\times n} \\ -\boldsymbol{M}_r^{-1}\boldsymbol{K}_e & -\boldsymbol{M}_r^{-1}(\boldsymbol{D}_r + \boldsymbol{B}_e) \end{pmatrix}. \tag{6.121}$$

Moreover, the matrix $\boldsymbol{A}$ must be recomputed at each control cycle since it depends on the linearization point of the tank dynamics and also on the dynamic parameters of the environment model. The output matrix $\boldsymbol{C} \in \mathbb{R}^{3n\times N}$ is also time-varying and depends on the contact state $\psi(t)$. In case $\psi(t) = 0$

$$\boldsymbol{C} = \begin{pmatrix} \boldsymbol{I}^{2n\times 2n} & 0^{2n\times 1} & -\boldsymbol{I}^{2n\times 2n} \\ 0^{n\times 2n} & 0^{n\times 1} & 0^{n\times 2n} \end{pmatrix} \tag{6.122}$$

while in case $\psi(t) = 1$

$$\boldsymbol{C} = \begin{pmatrix} \boldsymbol{I}^{2n\times 2n} & 0^{2n\times 1} & -\boldsymbol{I}^{n\times n} & -\boldsymbol{I}^{n\times n} \\ 0^{n\times 2n} & 0^{n\times 1} & -\boldsymbol{K}_e & -\boldsymbol{B}_e \end{pmatrix} \tag{6.123}$$

while the output disturbance matrix $\boldsymbol{D}^d \in \mathbb{R}^{3n\times 3n+1}$ is again time-varying but it does not depends on the contact state

$$\boldsymbol{D}^d = \begin{pmatrix} 0^{2n\times n} & 0^{2n\times n} & 0^{2n\times n} & 0 \\ 0^{n\times n} & 0^{n\times n} & \boldsymbol{K}_e & 0 \end{pmatrix}. \tag{6.124}$$

The output vector of such system

$$\boldsymbol{y}(t) = \begin{bmatrix} (\boldsymbol{x}_o - \boldsymbol{x}_r) \\ (\dot{\boldsymbol{x}}_o - \dot{\boldsymbol{x}}_r) \\ \boldsymbol{h}_r \end{bmatrix} \tag{6.125}$$

is composed of the position and velocity errors between the operator and remote robots, and the environment reaction force. The input matrix $\boldsymbol{B} \in \mathbb{R}^{N\times n}$ is

$$\boldsymbol{B} = \begin{cases} \boldsymbol{B}_o, & \text{MPC at the operator side} \\ \boldsymbol{B}_r, & \text{MPC at the remote side} \end{cases}$$

and the input disturbance matrix $\boldsymbol{B}^d \in \mathbb{R}^{N\times 3n+1}$ is

$$\boldsymbol{B}^d = \begin{cases} \boldsymbol{B}_o^d, & \text{MPC at the operator side} \\ \boldsymbol{B}_r^d, & \text{MPC at the remote side} \end{cases}$$

and they depend on the controlled manipulator. At the operator side, $\boldsymbol{B}_o$ and $\boldsymbol{B}_o^d$, are defined as

$$\boldsymbol{B}_o = \begin{pmatrix} 0^{n\times n} \\ \boldsymbol{M}_o^{-1} \\ -\bar{\dot{\boldsymbol{x}}}_o^T \\ 0^{n\times n} \\ 0^{n\times n} \end{pmatrix}, \quad \boldsymbol{B}_o^d = \begin{pmatrix} 0^{n\times n} & 0^{n\times n} & 0^{n\times n} & 0 \\ \boldsymbol{M}_o^{-1} & 0^{n\times n} & 0^{n\times n} & 0 \\ 0^{1\times n} & 0^{1\times n} & 0^{1\times n} & 1 \\ 0^{n\times n} & 0^{n\times n} & 0^{n\times n} & 0 \\ 0^{n\times n} & \boldsymbol{M}_r^{-1} & \boldsymbol{M}_r^{-1}\boldsymbol{K}_e & 0 \end{pmatrix} \tag{6.126}$$

On this side of the teleoperation control architecture, only the operator-side model is controlled while the remote-side model input is kept constant over the time horizon. At the remote side the input matrices $\boldsymbol{B}_r$ and $\boldsymbol{B}_r^d$ are

$$\boldsymbol{B}_r = \begin{pmatrix} 0^{n\times n} \\ 0^{n\times n} \\ -\bar{\dot{\boldsymbol{x}}}_r^T \\ 0^{n\times n} \\ \boldsymbol{M}_r^{-1} \end{pmatrix}, \quad \boldsymbol{B}_r^d = \begin{pmatrix} 0^{n\times n} & 0^{n\times n} & 0^{n\times n} & 0 \\ \boldsymbol{M}_o^{-1} & \boldsymbol{M}_o^{-1} & 0^{n\times n} & 0 \\ 0^{1\times n} & 0^{1\times n} & 0^{1\times n} & 1 \\ 0^{n\times n} & 0^{n\times n} & 0^{n\times n} & 0 \\ 0^{n\times n} & 0^{n\times n} & \boldsymbol{M}_r^{-1}\boldsymbol{K}_e & 0 \end{pmatrix} \tag{6.127}$$

and only the remote-side model is controlled by the MPC, while the operator-side manipulator input is kept constant over time. In the next two subsections, we will describe the MPC controllers at the operator and remote side. Is it worth noticing that to guarantee the passivity of the system we perform the check on the tank state variable $\hat{x}_t$ (i.e., $\hat{x}_t > \varepsilon$) instead of $E(t)$ (i.e., $E = \frac{1}{2}x_t^2 > \varepsilon$). This inequality is a linear constraint and is equivalent to the condition on the energy ($\hat{x}_t(t) > 0 \Leftrightarrow E(t) > 0, \forall t$), since $\hat{x}_t(t)$ is always positive as well.

Operator side teleoperation

Let $\boldsymbol{z}_o \in \mathbb{R}^N$ be the state vector of the MPC at the operator side

$$\boldsymbol{z}_o(k) = \begin{bmatrix} \boldsymbol{x}_o(k) \\ \dot{\boldsymbol{x}}_o(k) \\ x_{t_o}(k) \\ \boldsymbol{x}_r(k-\delta_o(k)) \\ \dot{\boldsymbol{x}}_r(k-\delta_o(k)) \end{bmatrix} \tag{6.128}$$

and let $f_o(\boldsymbol{z}, \boldsymbol{u}, \boldsymbol{d})$ be the update equation of the operator side MPC

$$\boldsymbol{z}_o(k+1) = f_o(\boldsymbol{z}_o(k), \boldsymbol{u}_o(k), \boldsymbol{d}_o(k)) = \bar{\boldsymbol{A}}\boldsymbol{z}_o(k) + \bar{\boldsymbol{B}}_o\boldsymbol{u}_o(k) + \bar{\boldsymbol{B}}_o^d\boldsymbol{d}_o(k) \tag{6.129}$$

where $\boldsymbol{x}_o$, $\dot{\boldsymbol{x}}_o$, x_{t_o} are the Cartesian position, velocity, and the tank value for the operator side, while $\boldsymbol{x}_r$, $\dot{\boldsymbol{x}}_r$ are the Cartesian position and velocity for

the remote side; $\boldsymbol{u}_o \in \mathbb{R}^n$ is the operator control command and $\delta_o(k) \in \mathbb{N}$ is the communication delay from environment to operator side at time $t = k\Delta t$. As before, the matrices $\bar{\boldsymbol{A}}$, $\bar{\boldsymbol{B}}$, $\bar{\boldsymbol{B}}^d$ are the discrete-time equivalent of the aforementioned continuous dynamic model. Due to the distributed approach the state of the opposite side dynamics and the last applied command are received after $\delta_o(k)$ steps of delay.

The operator side tank's level x_{t_o} is computed using the observed state $\boldsymbol{x}_o(k)$, $\dot{\boldsymbol{x}}_o(k)$ and the control input $\boldsymbol{u}_o(k)$. The remote side robot's model state $\boldsymbol{x}_r(k - \delta_o(k))$, $\dot{\boldsymbol{x}}_r(k - \delta_o(k))$ are estimated using the delayed control input $\boldsymbol{u}_r(k - \delta_o(k))$. The same works also for the estimation of the interaction torque $\hat{\boldsymbol{h}}_r$ which is computed on both sides of the teleoperation architecture based on the observed remote side robot state. The two estimators compute at every control cycle the tank level and the environment torque applying (5.27) and (6.13) respectively.

The cost function to be minimised would balance the position and velocity tracking error between the operator and remote sides. Moreover, the cost function moderates the control input $\hat{\boldsymbol{u}}_o$ and its rate of change $\Delta\hat{\boldsymbol{u}}_o$. The resulting quadratic cost function at time k is the same adopted for the non-linear MPC (6.24)

$$\begin{aligned}\mathcal{C}_o(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}_o) = &\sum_{i=1}^{k_p} \hat{\boldsymbol{e}}(k+i)^T \boldsymbol{\Lambda}_p \hat{\boldsymbol{e}}(k+i) + \hat{\dot{\boldsymbol{e}}}(k+i)^T \boldsymbol{\Lambda}_v \hat{\dot{\boldsymbol{e}}}(k+i) + \\ &+ \sum_{j=1}^{k_c} \hat{\boldsymbol{u}}_o(k+j)^T \boldsymbol{\Lambda}_c \hat{\boldsymbol{u}}_o(k+j) + \Delta\hat{\boldsymbol{u}}_o(k+j)^T \boldsymbol{\Lambda}_d \Delta\hat{\boldsymbol{u}}_o(k+j) \quad (6.130)\end{aligned}$$

The optimal control inputs $\hat{\boldsymbol{u}}_o^\star(k), \ldots, \hat{\boldsymbol{u}}_o^\star(k+k_c)$ are obtained as a solution of the finite-horizon constrained optimal LQ control problem

$$\begin{aligned}\hat{\boldsymbol{u}}_o^\star(k+i)|_{i=0}^{k_c} = \quad & \underset{\hat{\boldsymbol{u}}_o(\cdot)}{\arg\min} \quad \mathcal{C}_o(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}_o, \boldsymbol{d}_o) \\ & \text{subject to} \quad \hat{\boldsymbol{z}}(k) = \boldsymbol{z}(k) \\ & \qquad\qquad\quad \hat{\boldsymbol{z}}(k+1) = f_o(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}_o, \boldsymbol{d}_o) \\ & \qquad\qquad\quad \hat{\boldsymbol{z}}(k+k_p) \in \mathcal{Z}_o^f \\ & \qquad\qquad\quad \hat{x}_{T_o} \geq \varepsilon_o \\ & \qquad\qquad\quad (\boldsymbol{u}_o(k), \dot{\boldsymbol{x}}_o(k)) \in \mathcal{P}_\sigma^o\end{aligned}$$

where

$$\begin{aligned}\mathcal{Z}_o^f = \{(\hat{\boldsymbol{x}}_o, \hat{\dot{\boldsymbol{x}}}_o, \hat{\boldsymbol{x}}_r, \hat{\dot{\boldsymbol{x}}}_r) | \hat{\boldsymbol{x}}_o = \hat{\boldsymbol{x}}_r = \boldsymbol{x}_r(k - \delta_o(k) + k_p) \\ \wedge \hat{\dot{\boldsymbol{x}}}_o = \hat{\dot{\boldsymbol{x}}}_r = \dot{\boldsymbol{x}}_r(k - \delta_o(k) + k_p)\} \quad (6.131)\end{aligned}$$

is the terminal constraint on the robot's dynamics. We also assume $\boldsymbol{u}_r$ and $\boldsymbol{h}_o$ constants over the optimisation horizon.

Remote side teleoperation

For the remote side, the optimisation problem is similar as the operator side. The state vector $\boldsymbol{z}_r \in \mathbb{R}^N$ has the same dimension of $\boldsymbol{z}_o$ and is given by

$$\boldsymbol{z}_r(k) = \begin{bmatrix} \boldsymbol{x}_o(k-\delta_r(k)) \\ \dot{\boldsymbol{x}}_o(k-\delta_r(k)) \\ x_{t_r}(k) \\ \boldsymbol{x}_r(k) \\ \dot{\boldsymbol{x}}_r(k) \end{bmatrix} \tag{6.132}$$

where $\delta_r(k) \in \mathbb{N}$ is the communication delay from the remote side to the operator side. The update equation for $\boldsymbol{z}_r$ and the cost function (6.130) must be changed accordingly with the new state vector $\boldsymbol{z}_r$ and with the new control input $\boldsymbol{u}_r$. Moreover, the optimisation problem is subjected to a new additional constraint since the controller at the remote side must enforce the interaction force upper-bound (6.133)

$$\begin{aligned} \hat{\boldsymbol{u}}_r^\star(k+i)|_{i=0}^{k_c} = \quad & \underset{\hat{\boldsymbol{u}}_r(\cdot)}{\arg\min} \quad \mathcal{C}_r(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}_r, \boldsymbol{d}_r) \\ & \text{subject to} \quad \hat{\boldsymbol{z}}(k) = \boldsymbol{z}(k) \\ & \qquad \hat{\boldsymbol{z}}(k+1) = f_r(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}_r, \boldsymbol{d}_r) \\ & \qquad \hat{\boldsymbol{z}}(k+k_p) \in \mathcal{Z}_r^f \\ & \qquad \hat{x}_{T_r} \geq \varepsilon_r \\ & \qquad (\boldsymbol{u}_r(k), \dot{\boldsymbol{x}}_r(k)) \in \mathcal{P}_\sigma^r \\ & \qquad \hat{\boldsymbol{h}}_r \leq \mathcal{T}^{max} \qquad (6.133) \\ & \qquad \hat{\boldsymbol{h}}_r \geq 0. \qquad (6.134) \end{aligned}$$

where

$$\begin{aligned} \mathcal{Z}_r^f = \{(\hat{\boldsymbol{x}}_o, \hat{\dot{\boldsymbol{x}}}_o, \hat{\boldsymbol{x}}_r, \hat{\dot{\boldsymbol{x}}}_r) | \hat{\boldsymbol{x}}_o = \hat{\boldsymbol{x}}_r = \boldsymbol{x}_o(k-\delta_r(k)+k_p) \\ \wedge \hat{\dot{\boldsymbol{x}}}_o = \hat{\dot{\boldsymbol{x}}}_r = \dot{\boldsymbol{x}}_o(k-\delta_r(k)+k_p)\}. \end{aligned} \tag{6.135}$$

Since the environment model is constant over the prediction horizon, the contact state cannot change as well (6.134).

6.6 Passivity-based hybrid linear MPC

The non-linear and linear P-MPC presented in the previous sections are on one side a slow but precise model of the whole teleoperation system, and on

the other side faster but approximated. The main limitation on the linear approximation is the assumption of having a stable contact state during the prediction horizon. This is due to the linear approximation of the interaction dynamics. To cope with this limitation we propose to model the teleoperation as a hybrid system and we will present a hybrid linear P-MPC, which can be seen as a trade-off solution between the methodologies proposed above.

In general, hybrid systems are a mixture of continuos time dynamics and discrete events. As opposed to classical control design where the event-driven and time-driven dynamics are considered separately, in hybrid systems such dynamics interact. Such separation of asynchronous and synchronous controls will in many cases, however, lead to a too-conservative design. The interaction between these two dynamics are such that changes in the system state occur both in response of discrete events and continuos dynamics, described by difference and differential equation, respectively. A hybrid control system is a control system, where the plant or the controller, individually, or in combination, can be modelled as a hybrid system.

Let the dynamic model of a manipulator in the operational space be

$$\boldsymbol{M}(\boldsymbol{x})\ddot{\boldsymbol{x}} + \boldsymbol{C}(\dot{\boldsymbol{x}}, \boldsymbol{x})\dot{\boldsymbol{x}} + \boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{h} + \boldsymbol{u} \tag{6.136}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ are the Cartesian space coordinates of the robot, $\boldsymbol{M}(\boldsymbol{x})$, $\boldsymbol{C}(\dot{\boldsymbol{x}}, \boldsymbol{x})$, $\boldsymbol{g}(\boldsymbol{x})$ are the inertia, Coriolis and gravity terms, $\boldsymbol{h}$ and $\boldsymbol{u}$ are the external and the control forces/torques. As in Section 6.2, the robot model used within the hybrid-MPC is the exact linearisation via feedback of the robot dynamics defined in (6.136).

6.6.1 Linearized energy tank

As we pointed out in the previous section, the energy-storing system defined in (5.27)

$$\dot{x}_{t_w} = \frac{\dot{\boldsymbol{x}}_w^T \boldsymbol{D}_w \dot{\boldsymbol{x}}_w + \sigma_w P_w^{in}(t)}{x_{t_w}} - \frac{P_w^{out}(t)}{x_{t_w}} + u_{t_w} \tag{6.137}$$

is non-linear since the energy must be computed at the power port of the tank and this prevents the definition of a quadratic optimization problem within the MPC. For this reason we adopt its first-order linear approximation as defined in Section 6.5.1 and so also in this case to guarantee the passivity of the system we perform the check on the tank state variable x_t.

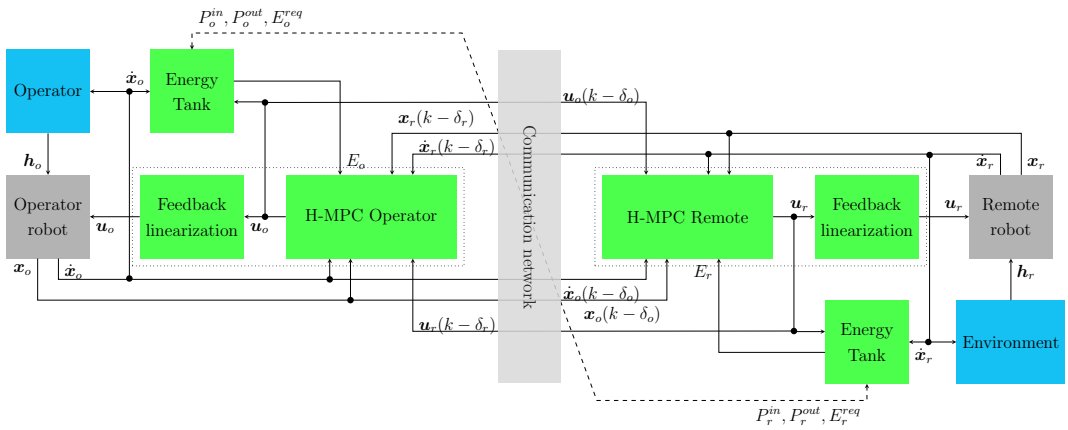


Figure 6.17: The MPC-based control architecture. The green blocks are the H-MPC control modules, the blue blocks are the external torque sources, the grey blocks are the manipulators and the light grey is the communication channel.

6.6.2 Hybrid bilateral teleoperation

Since the teleoperation control law is distributed as shown in Figure 6.17, the control system must be replicated on both sides of the communication channel. The model is not constant over time since the local linearization of the tank and the estimation of the environment should be recomputed at each control cycle. Moreover, even if the general structure of the hybrid systems is the same on both sides of the teleoperation architecture, the actual operator/remote systems are subjected to different exogenous inputs and have different matrices.

Let $(\boldsymbol{z}, q, b)$ be the state vector of the hybrid system modelling the bilateral teleoperation: similarly to the previous section, $\boldsymbol{z} \in \mathbb{R}^N$, with $N = 4n + 1$, is the state of the two manipulators and the energy tank of the controlled side. The contact state at the remote side is represented by the logic variable $q \in Q = \{0, 1\}$, while the number of environment transitions is $b \in \mathbb{N}$. The continuous dynamics is

$$\begin{cases} \dot{\boldsymbol{z}}(t) = \boldsymbol{A}(q)\boldsymbol{z}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{B}^d\boldsymbol{d}(t) \\ \dot{q}(t) = 0 \\ \dot{b}(t) = 0 \end{cases} \tag{6.138}$$

where $(\boldsymbol{z}, q, b) \in C := \mathbb{R}^N \times Q \times \mathbb{N}$ and $\boldsymbol{d}(t)$ is a measured exogenous input similar to the ones defined in (6.115) and (6.116) for the operator and remote side respectively. The main difference is that in the case of hybrid system the

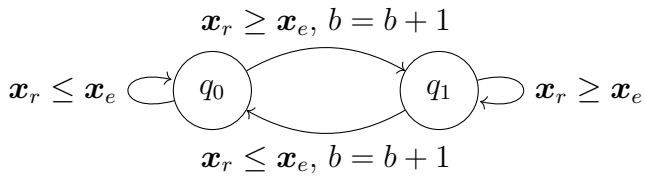


Figure 6.18: The finite state machine representing the jump dynamics of the hybrid bilateral teleoperation.

switching behaviour of $\psi(t)\boldsymbol{x}_e$ is modelled within (6.138) such that

$$\boldsymbol{d}_o = \begin{bmatrix} \boldsymbol{h}_o \\ \boldsymbol{u}_r \\ q\boldsymbol{x}_e \\ (\dot{\bar{\boldsymbol{x}}}_o^T \bar{\boldsymbol{u}}_o + \bar{p}_o) \end{bmatrix} \tag{6.139}$$

while at the remote side is defined as

$$\boldsymbol{d}_r = \begin{bmatrix} \boldsymbol{h}_o \\ \boldsymbol{u}_o \\ q\boldsymbol{x}_e \\ (\dot{\bar{\boldsymbol{x}}}_r^T \bar{\boldsymbol{u}}_r + \bar{p}_r) \end{bmatrix}. \tag{6.140}$$

As shown in Figure 6.18, the interaction with the environment is modelled as a two-state automaton where the current state $q \in \{q_0, q_1\}$ selects the appropriate continuous dynamics:

- q_0 means that the remote manipulator is in free-motion
- q_1 means that the remote manipulator is in contact.

When the current state q goes from $0 \to 1$ or $1 \to 0$ the number of transitions b is increased by one. The jump dynamics is given by

$$\boldsymbol{z}^+ = \boldsymbol{z}, \quad q^+ = 1, \quad b^+ = b + 1, \qquad (\boldsymbol{z}, q, b) \in D_0 \tag{6.141}$$

$$\boldsymbol{z}^+ = \boldsymbol{z}, \quad q^+ = 0, \quad b^+ = b + 1, \qquad (\boldsymbol{z}, q, b) \in D_1 \tag{6.142}$$

where the jump set is $D = D_0 \cup D_1$

$$D_0 = \{(\boldsymbol{z}, q, b) \in \mathbb{R}^N \times Q \times \mathbb{N} \mid q = 0 \wedge \boldsymbol{x}_r \geq \boldsymbol{x}_e\}, \tag{6.143}$$

$$D_1 = \{(\boldsymbol{z}, q, b) \in \mathbb{R}^N \times Q \times \mathbb{N} \mid q = 1 \wedge \boldsymbol{x}_r \leq \boldsymbol{x}_e\}. \tag{6.144}$$

The output map of the hybrid system is the output vector (6.125) together with the number of state transitions b

$$\boldsymbol{y}(t) = \begin{bmatrix} \boldsymbol{C}(q)\boldsymbol{z}(t) + \boldsymbol{D}^d \boldsymbol{d}(t) \\ b \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_o - \boldsymbol{x}_r \\ \dot{\boldsymbol{x}}_o - \dot{\boldsymbol{x}}_r \\ \boldsymbol{h}_r \\ b \end{bmatrix} \tag{6.145}$$

Continuous dynamics

The continuous dynamic of the hybrid system can be directly derived from the previous section. In fact, the matrices which previously was defined depending on $\psi(t)$ now are function of the contact state q.

The state evolution matrix $\boldsymbol{A}(q) \in \mathbb{R}^{N \times N}$ at the operator side results to be

$$\boldsymbol{A}_o(q) = \begin{pmatrix} \boldsymbol{A}_{1,1} & 0^{2n\times 1} & 0^{2n\times 2n} \\ 0^{1\times n} - \bar{\boldsymbol{u}}_o^T & 0^{1\times 1} & 0^{1\times 2n} \\ 0^{2n\times 2n} & 0^{2n\times 1} & \boldsymbol{A}_{2,2}(q) \end{pmatrix} \tag{6.146}$$

while at the remote side is

$$\boldsymbol{A}_r(q) = \begin{pmatrix} \boldsymbol{A}_{1,1} & 0^{2n\times 1} & 0^{2n\times 2n} \\ 0^{1\times 2n} & 0^{1\times 1} & 0^{1\times n} - \bar{\boldsymbol{u}}_r^T \\ 0^{2n\times 2n} & 0^{2n\times 1} & \boldsymbol{A}_{2,2}(q) \end{pmatrix} \tag{6.147}$$

The operator side robot model is the same as before, while the remote side robot model $\boldsymbol{A}_{2,2}(q)$ can be rewritten as

$$\boldsymbol{A}_{2,2}(q) = \begin{pmatrix} 0^{n\times n} & \boldsymbol{I}^{n\times n} \\ -q\boldsymbol{M}_r^{-1}\boldsymbol{K}_e & -\boldsymbol{M}_r^{-1}(\boldsymbol{D}_r + q\boldsymbol{B}_e) \end{pmatrix}. \tag{6.148}$$

Similar to $\boldsymbol{A}$, the output matrix $\boldsymbol{C}(q) \in \mathbb{R}^{3n\times N}$ can be rewritten as

$$\boldsymbol{C}(q) = \begin{pmatrix} \boldsymbol{I}^{2n\times 2n} & 0^{2n\times 1} & -\boldsymbol{I}^{n\times n} & -\boldsymbol{I}^{n\times n} \\ 0^{n\times 2n} & 0^{n\times 1} & -q\boldsymbol{K}_e & -q\boldsymbol{B}_e \end{pmatrix} \tag{6.149}$$

The other matrices $\boldsymbol{B}_o$, $\boldsymbol{B}_r$, $\boldsymbol{B}_o^d$, $\boldsymbol{B}_r^d$, $\boldsymbol{D}^d$ which define the model of the bilateral teleoperation are the same as in the previous section.

6.6.3 Hybrid MPC

The optimal control strategies adopted at the operator and remote sides are based on the design of hybrid-MPC for piece-wise affine (PWA) systems. The discretized version of the hybrid system in (6.138) can be translated into an equivalent PWA system [175] and this process is fully automated using the HYbrid System DEscription Language (HYSDEL) framework [176].

Operator side teleoperation

Let $(\boldsymbol{z}_o, q_o, b_o)$ be the state vector of the hybrid-MPC at the operator side. Based on the PWA representation of the hybrid bilateral teleoperation system, the hybrid-MPC is computed using the multi-parametric constrained receding

horizon control design for hybrid systems [177]. We use the quadratic norm as metric and we minimize a cost function similar to (6.24)

$$\begin{aligned}\mathcal{C}_o(\hat{\boldsymbol{z}}, \hat{\boldsymbol{u}}_o) = \sum_{i=1}^{k_p} & \hat{\boldsymbol{e}}(k+i)^T \boldsymbol{\Lambda}_p \hat{\boldsymbol{e}}(k+i) + \hat{\dot{\boldsymbol{e}}}(k+i)^T \boldsymbol{\Lambda}_v \hat{\dot{\boldsymbol{e}}}(k+i) + \\ & + \Delta\hat{\boldsymbol{u}}_o(k+i)^T \boldsymbol{\Lambda}_d \Delta\hat{\boldsymbol{u}}_o(k+i) + \\ & + \lambda b_o(k+i) \end{aligned} \tag{6.150}$$

where $\boldsymbol{\Lambda}_p$, $\boldsymbol{\Lambda}_v$ and $\boldsymbol{\Lambda}_d$ are positive semi-definite weight matrices and λ is the scalar weight for the number of contact transitions. The optimization is subjected to the following inequality constraints

$$\hat{x}_{t_o} \geq \varepsilon_o \tag{6.151}$$

$$\hat{x}_{t_r} \geq \varepsilon_r \tag{6.152}$$

which guarantee together with the under-approximation constraints defined in the previous section that the overall teleoperation system is passive by construction.

Remote side teleoperation

For the remote side, the optimisation problem is similar as the operator side with the following state vector $(\boldsymbol{z}_r, q_r, b_r)$ where now the components of $\boldsymbol{z}_r$ of the operator side are initialized with $\delta_r(k) \in \mathbb{N}$ steps of communication delay. The update equation for $\boldsymbol{z}_r$ and the cost function (6.150) must be changed accordingly to the new definition of $\boldsymbol{z}_r$, the number of transitions b_r and with the new control input $\boldsymbol{u}_r$. Moreover, the optimisation problem is subjected to additional constraints since the controller at the remote side must enforce the interaction force bound (6.153)

$$\begin{aligned} \hat{x}_{t_o} &\geq \varepsilon_o \\ \hat{x}_{t_r} &\geq \varepsilon_r \\ \hat{\boldsymbol{h}}_r &\leq \mathcal{T}^{max}. \end{aligned} \tag{6.153}$$

6.6.4 Simulation results

The proposed method has been validated in a simulated environment using the *Hybrid Toolbox* [176] with *Matlab 2021a*. As in Section 6.3.4 we modelled the behaviour of the human operator (i.e., $\boldsymbol{h}_o$) as a PD position controller. The two hybrid-MPCs provide as in the other teleoperation scenarios, the

force feedback and the command torques to the operator and remote side manipulators, respectively.

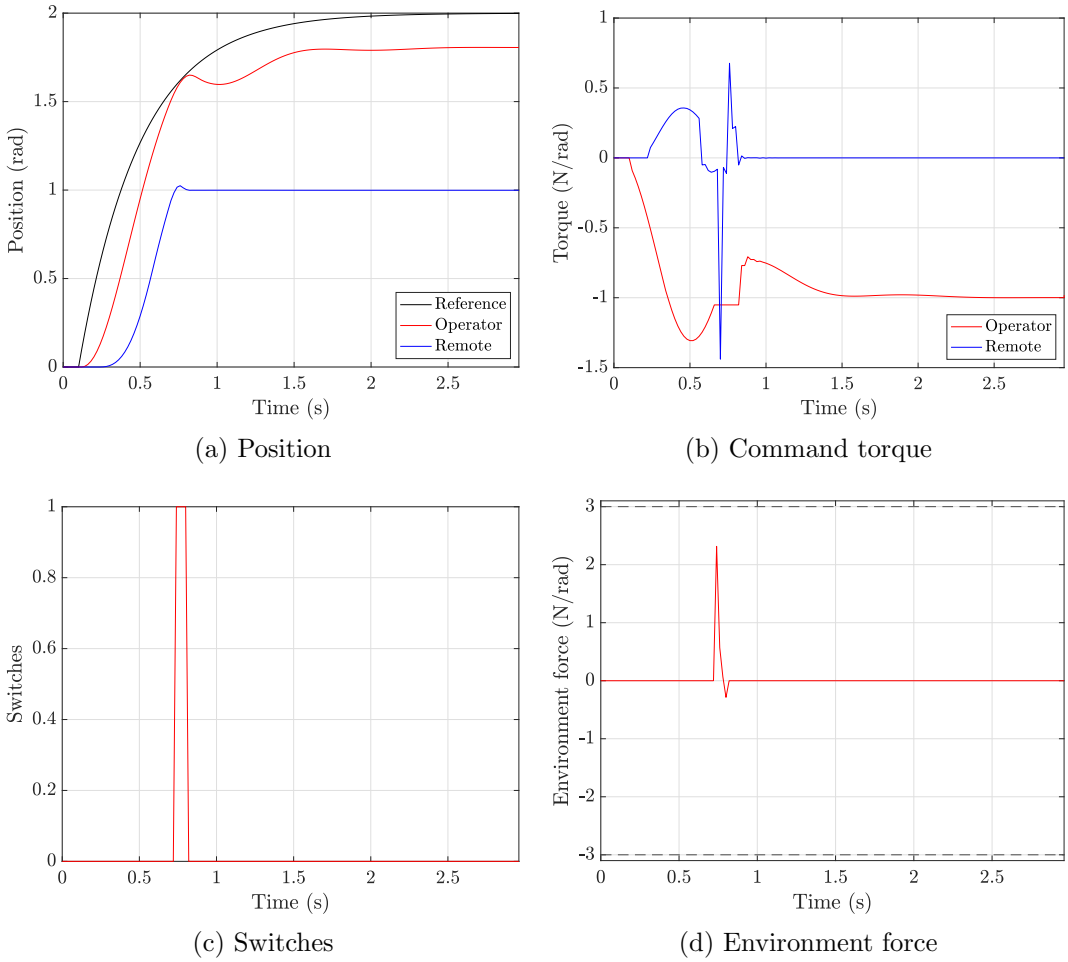
The simulated teleoperation setup consists of two 1-DOF robotic manipulators and a communication channel with a constant delay of 0.1 s in both directions. The sampling time of the controller is set to $\Delta t = 0.02$ s and the prediction horizon of the hybrid-MPC equals to 8 steps. The thresholds of the tanks are the same for the both sides of the teleoperation and are set to $T^{max} = 5$, $T^{ava} = 0.6$, $T^{req} = 0.2$, $\epsilon = 0.001$, $\bar{P} = 0.01$. The tanks are initialised to $T^{max}/2$ for the operator side and $T^{max}/2$ for the remote side. The robot parameters, inertia $j = 0.0266\,\mathrm{kgm}^2$ and friction $d = 0.0218\,\mathrm{N\,s}$, are the same for both motors. Since the environment parameters are estimated online, to provide a conservative control action during the convergence of the estimator, we initialize the control system with an overestimation. In our setup this estimation has been empirically set to be the 110% of the nominal parameters.

Soft contact with force constraint

In the first scenario, we simulate contact with a soft material placed 1 rad ahead of the starting position of the remote-side manipulator. The nominal parameters of the environment model are $K_e = 500$ and $B_e = 10$. The interaction force is constrained to be ≤ 3 N, which is at least a reduction of the 60% of the unconstrained peak force. Figure 6.19a shows the desired position for the operator together with the actual position of the operator (i.e., haptic device) and the remote robot. The contact with the environment occurs between 0.5 s to 1 s and the system quickly reaches the equilibrium. The control commands at both sides of the teleoperation are shown in Figure 6.19b. The action of the controller can prevent any bounce-off effect as shown in Figure 6.19c. Finally, Figure 6.19d shows the evolution of the environment reaction force. The combination of bounce reduction and force constraints guarantees the force to be correctly bounded within the desired upper limit.

Hard contact with bounce reduction

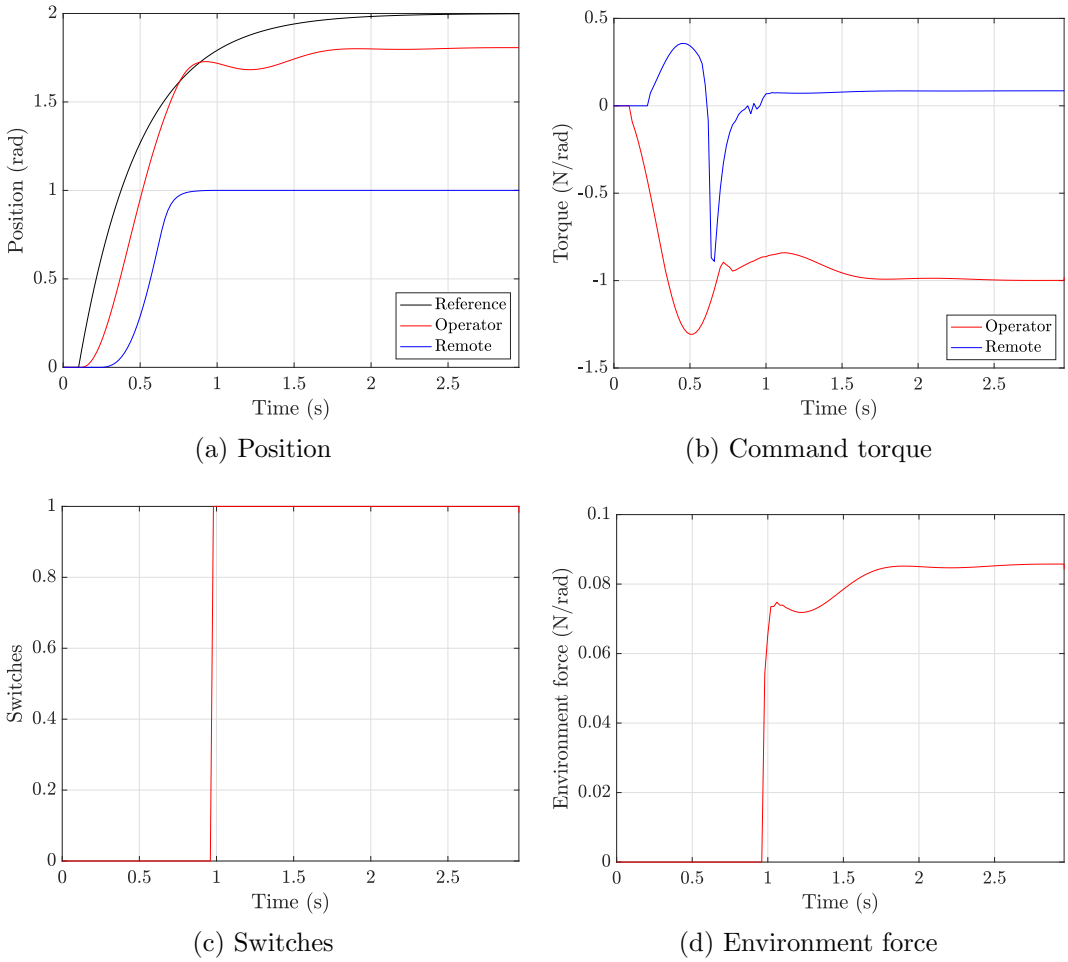
In the second scenario, we simulate contact with a hard material placed again at 1 rad ahead of the remote-side manipulator's starting position. For the hard contact, the nominal parameters of the environment are $K_e = 2500$ and $B_e = 12$. As in the previous scenario the positions of the two manipulators, Figure 6.20a, do not show oscillatory behaviour during the transition between free-motion and contact. And again, the system can reach equilibrium in about 2 s. The command torques, shown in Figure 6.20b, are smooth despite the stiff



(a) Position (b) Command torque

(c) Switches (d) Environment force

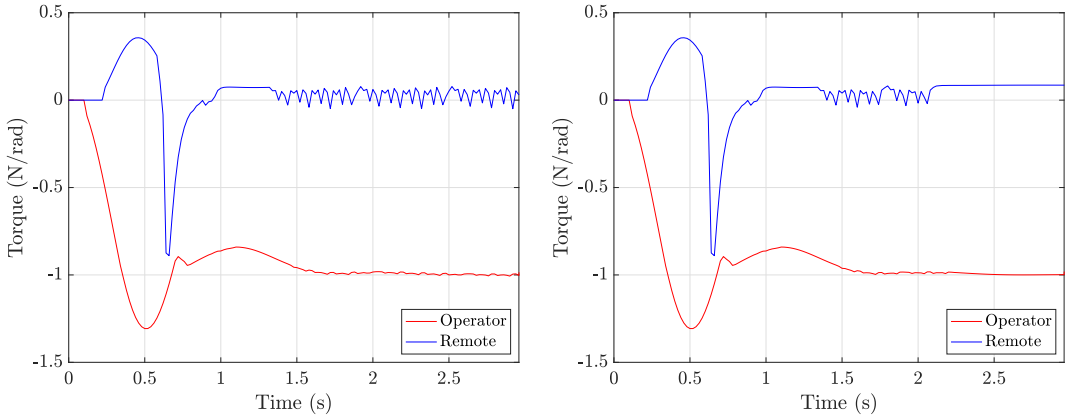
Figure 6.19: Simulation of soft contact with force constraints under constant communication delay. a) in black the desired trajectory, and in red and blue the position of the operator- and remote-side manipulators, respectively, b) command torques at the operator side (red) and at the remote-side (blue), c) switches of the contact state over time, d) the reaction environment force, in dashed black the upper and lower bounds.



(a) Position (b) Command torque

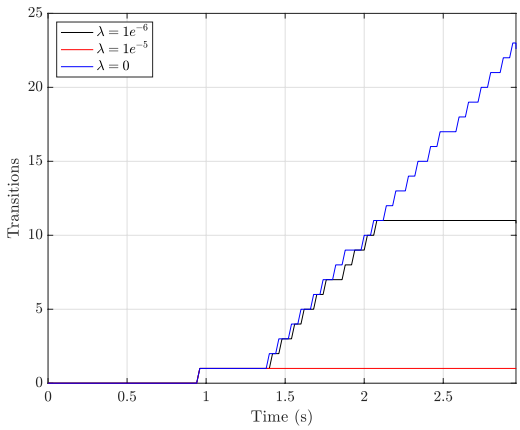
(c) Switches (d) Environment force

Figure 6.20: Simulation of hard contact with bounce reduction under constant communication delay. a) in black the desired trajectory, and in red and blue the position of the operator- and remote-side manipulators, respectively, b) command torques at the operator side (red) and at the remote-side (blue), c) switches of the contact state over time, d) the reaction environment force.



(a) Command torque ($\lambda = 0$)

(b) Command torque ($\lambda = 1e^{-6}$)



(c) Number of switches

Figure 6.21: Effect of the bounces reduction. a) the chattering on the command input with $\lambda = 0$, b) the chattering effect on the command input with $\lambda = 1e^{-6}$, c) the amount of contact state transitions in function of the λ weight of the cost function.

environment. Moreover, as shown in Figure 6.20c, the proposed method does not cause any chattering in the transition between free motion and contact. As opposed to the soft contact scenario, the environment force is always close to zero, see Figure 6.20d. This behaviour is caused by the ratio between the weight on the control input variation and the weight on the tracking error. Since the system reaches the equilibrium with a very small indentation within the environment, the remote-side controller could be subjected to chattering and so the reduction of the contact transitions plays an important role. For this reason, in the same scenario, we performed multiple experiments varying the weight λ to see how its value in the cost function affects the amount of chattering. As shown in Figure 6.21, with $\lambda = 0$, i.e., no penalty on b, the number of contact transitions is quite high. While with $\lambda > 0$ the number of transitions is dramatically reduced.

Chapter 7

Conclusion and Future Works

In this thesis, we aimed to investigate the applications of optimal control for a semi-autonomous robotic surgical system. We exploit optimal control to manage the unexpected events which would eventually occur during a surgical procedure. We focus on designing a novel constrained bilateral teleoperation, where safety conditions are guaranteed even in the case of manual intervention. The experimental validation of the proposed methodologies has been carried out within the *SARAS* project and has been conducted on a synthetic anatomical phantom. As described in Chapter 2, the experimental setup consists of the da Vinci® robotic platform and the custom *SARAS* robotic arms.

In Chapter 3, we presented and tested the preliminary technologies used in Chapter 4 to implement the robotic system's supervisory controller. Such technologies are a novel 3D calibration procedure for a multi-arm robotic system and a point cloud-based registration for pre-operative data. The proposed calibration outperformed the state-of-the-art reaching an accuracy below 1 mm during a dual-arm manipulation scenario. The main drawback of this calibration technique is the current limitation of RGB-D camera, which nowadays limits its application in surgery due to the lack of RGB-D endoscope. Anyhow, the proposed methodology can be extended to the standard stereo endoscope. Another improvement is to automate the calibration procedure which can significantly simplifies its implementation in a realistic surgical setup. The proposed pre-operative registration routine is able to map a 3D anatomical model within the patient's anatomy exploiting a feature-based 3D aligning with a sparse point cloud from SLAM. We also provided an effective solution to derive the scaling factor of the SLAM reconstruction using robot kinematics. The proposed registration currently does not take into account any environment deformation and therefore can map only static information; we plan to extend it also to dynamic objects in the future. To do so, we plan to update

the pre-operative model with the deformation and perform the registration online with the extended SLAM.

In Chapter 4, we presented the modelling framework used to define and implement the supervisory controller adopted in this thesis. The proposed methodology, together with the previous calibration and registration techniques, proved to be robust, modular, and flexible. We successfully executed the tasks usually performed by an assistant surgeon in a semi-autonomous fashion. The system successfully assisted the main surgeon on the RARP, and, during the extensive verification of the platform, we saw only 2 phases out of 38 interrupted by an unrecoverable error. Such situations were caused by inappropriate initial positioning of the pre-operative points of interest and the surgeon's inexperience while collaborating with *SARAS* arms.

In Chapter 5, we presented the bilateral teleoperation technique and we proposed some improvements to the *two-layer* control methodology. Firstly, we improved the energy tank dynamics by implementing a more efficient energy sharing protocol. Then, we proposed a revised version of the *two-layer* bilateral teleoperation algorithm adopting an optimal modulation policy. The new energy tank properly implements the upper-bound and provides a better exchange of the exceeding energy between the two sides of the teleoperation system. This new methodology proven to reduce conservativeness and improve transparency while guaranteeing passivity. We later coupled the energy tank and the manipulator exploiting a convex optimization problem to ensure robust stability and optimal performance. The optimal modulation had been experimentally validated both on a KUKA manipulator and the da Vinci® robotic system, while the new energy tank dynamics has been numerically verified in simulation.

In Chapter 6, we recast the *two-layer* approach in a passivity-based optimal control problem. In particular, we initially defined a non-linear P-MPC to constrain the interaction forces and then we proposed a linearized version of the P-MPC controller to provide real-time capabilities for multi-DOF manipulators. Finally, we presented an initial analysis of the application of hybrid-system theory to model the switching interaction between the robot's end-effector and the environment. The proposed P-MPCs demonstrated the capability of always guaranteeing the stability of the teleoperation also under communication delay. To reduce the computational effort we run the P-MPC on top of a feedback linearization of the robot dynamics. We also show that in case of position-controlled manipulators, the methodology can be used by considering also the controller dynamics inside the feedback. Moreover, we proposed and proved the effectiveness of a linearized energy tank dynamics

allowing us to embed the passivity constraint for linear P-MPC. In the end, the hybrid linear P-MPC demonstrated smooth transitions between contact and free motion while guaranteeing passivity and enforcing force constraints The methodology has been able to mitigate and in some cases prevent the bouncing-off effect induced by the interaction with the environment. In the future we plan to integrate and experimental validate the proposed linear P-MPC in the Franka Emika robots within the *SARAS* platforms.

Appendix A

libmpc++

libmpc++ is a free/open-source library for solving *linear* and *non-linear* Model Predictive Control (MPC). The library is written in standard C++17 and provides static and dynamic memory allocation via templated interfacing classes. It is available on Linux, MacOs and Windows and comes with a limited set of dependencies. It provides:

- Support for linear and non-linear MPC optimal control problem formulation
- Handles discrete-time and continuos-time (for the non-linear MPC only) system's dynamics definition
- Different length for the prediction and control horizon
- Automatic Jacobian approximation for non-linear MPC
- Header-only implementation
- Free/open-source software

To solve the optimization problems, libmpc++ uses OSQP [172] and NLopt [178] for linear and non-linear MPC, respectively. The source code and the references are available here `https://github.com/nicolapiccinelli/libmpc`.

A.1 Linear MPC

The linear MPC address the solution of the following convex quadratic optimization problem

$$\begin{aligned}
\text{minimize} \quad & \textstyle\sum_{k=0}^{N}(\boldsymbol{y}_k - \boldsymbol{y}_r)^T \Lambda_y (\boldsymbol{y}_k - \boldsymbol{y}_r) + \sum_{k=0}^{N-1}(\boldsymbol{u}_k - \boldsymbol{u}_r)^T \Lambda_u (\boldsymbol{u}_k - \boldsymbol{u}_r) + \\
& + \textstyle\sum_{k=0}^{N-1}(\Delta\boldsymbol{u}_k - \Delta\boldsymbol{u}_r)^T \Lambda_{du} (\Delta\boldsymbol{u}_k - \Delta\boldsymbol{u}_r) \\
\text{subject to} \quad & \boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + B\boldsymbol{u}_k + B^d\boldsymbol{d}_k \\
& \boldsymbol{x}_{\min} \leq \boldsymbol{x}_k \leq \boldsymbol{x}_{\max} \\
& \boldsymbol{y}_{\min} \leq \boldsymbol{y}_k \leq \boldsymbol{y}_{\max} \\
& \boldsymbol{u}_{\min} \leq \boldsymbol{u}_k \leq \boldsymbol{u}_{\max} \\
& s_{\min} \leq \bar{\boldsymbol{x}}^T\boldsymbol{x}_k + \bar{\boldsymbol{u}}^T\boldsymbol{u}_k \leq s_{\max}
\end{aligned} \tag{A.1}$$

where the states $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$, the outputs $\boldsymbol{y}_k \in \mathbb{R}^{n_y}$ and the inputs $\boldsymbol{u}_k \in \mathbb{R}^{n_u}$ are constrained to be between some lower and upper bounds. The problem is solved repeatedly for varying initial state $\boldsymbol{x}_0 \in \mathbb{R}^{n_x}$ and the scalar multipliers $\bar{\boldsymbol{x}} \in \mathbb{R}^{n_x}$ and $\bar{\boldsymbol{u}} \in \mathbb{R}^{n_u}$. The underlying linear system used within the MPC is defined as

$$\begin{cases} \boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + B\boldsymbol{u}_k + B^d\boldsymbol{d}_k \\ \boldsymbol{y}_k = C\boldsymbol{x}_k + D^d\boldsymbol{d}_k \end{cases} \tag{A.2}$$

where $\boldsymbol{d}_k \in \mathbb{R}^{n_{du}}$ is an additional exogenous input.

A.2 Non-linear MPC

The non-linear MPC address the solution of a generic non-linear optimization problem

$$\begin{aligned}
\text{minimize} \quad & \mathcal{C}(\boldsymbol{x}_k, \boldsymbol{u}_k) \\
\text{subject to} \quad & \boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k) \\
& l(\boldsymbol{x}_k, \boldsymbol{y}_k, \boldsymbol{u}_k) \leq 0 \\
& h(\boldsymbol{x}_k, \boldsymbol{u}_k) = 0 \\
& \boldsymbol{x}_0 = \bar{x}
\end{aligned} \tag{A.3}$$

where the states $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$, the outputs $\boldsymbol{y}_k \in \mathbb{R}^{n_y}$ and the inputs $\boldsymbol{u}_k \in \mathbb{R}^{n_u}$ can be arbitrary constrained by defining the function $l(\boldsymbol{x}_k, \boldsymbol{y}_k, \boldsymbol{u}_k)$ and $h(\boldsymbol{x}_k, \boldsymbol{y}_k, \boldsymbol{u}_k)$. The problem is solved repeatedly for varying initial state $\bar{x} \in \mathbb{R}^{n_x}$ and minimizing the user defined objective function $\mathcal{C}(\boldsymbol{x}_k, \boldsymbol{u}_k)$. The underlying non-linear system used within the MPC is defined as

$$\begin{cases} \boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k) \\ \boldsymbol{y}_k = g(\boldsymbol{x}_k, \boldsymbol{u}_k) \end{cases} \tag{A.4}$$

in case of continuos time system the function $\dot{x} = f(x, u)$ should be interpreted as the vector field of the desired dynamical system.

A.3 User manual

In the following section we will present a brief introduction on how to use libmpc++, the tutorials are just to give you an overview on the main part of the library.

A.3.1 Allocation

The main classes you deal with are the API interface to the linear and non-linear MPC, called **LMPC** and **NLMPC** respectively. Depending on the allocation type desired there are two ways of creating these objects, one by using templates and one by providing the dimensions as constructor arguments. The first can be used when the dimensions of the MPC problem are know at the compile time and the latter in the other case.

Static allocation syntax

```
mpc::NLMPC<Tnx, Tnu, Tny, Tph, Tch, Tineq, Teq> nlmpc;
mpc::LMPC<Tnx, Tnu, Tndu, Tny, Tph, Tch> lmpc;
```

Dynamic allocation syntax

```
mpc::NLMPC<> nlmpc(Tnx, Tnu, 0, Tny, Tph, Tch, Tineq, Teq);
mpc::LMPC<> lmpc(Tnx, Tnu, Tndu, Tny, Tph, Tch, 0, 0);
```

A.3.2 Solver parametrization

The inner solvers can be parametrized by using the following structures:

Non-linear MPC solver (nlopt)

```
NLParameters params;

params.relative_ftol = 1e-10;
params.relative_xtol = 1e-10;
params.hard_constraints = true;

nlmpc.setOptimizerParameters(params);
```

Linear MPC solver (OSQP)

```
LParameters params;

params.alpha = 1.6;
params.rho = 1e-6;
params.eps_rel = 1e-4;
params.eps_abs = 1e-4;
params.eps_prim_inf = 1e-3;
params.eps_dual_inf = 1e-3;
params.verbose = false;
params.adaptive_rho = true;
params.polish = true;

lmpc.setOptimizerParameters(params);
```

A.3.3 Example of linear MPC

This example shows how to regulate a quadcopter about a reference state with constrained control input and state space

```
lmpc.setLoggerLevel(mpc::Logger::log_level::NORMAL);

mpc::mat<Tnx, Tnx> Ad;
Ad << 1, 0, 0, 0, 0, 0, 0.1, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0.1, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0.1, 0, 0, 0,
0.0488, 0, 0, 1, 0, 0, 0.0016, 0, 0, 0.0992, 0, 0,
0, -0.0488, 0, 0, 1, 0, 0, -0.0016, 0, 0, 0.0992, 0,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0.0992,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0.9734, 0, 0, 0, 0, 0, 0.0488, 0, 0, 0.9846, 0, 0,
0, -0.9734, 0, 0, 0, 0, 0, -0.0488, 0, 0, 0.9846, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.9846;

mpc::mat<Tnx, Tnu> Bd;
Bd << 0, -0.0726, 0, 0.0726,
-0.0726, 0, 0.0726, 0,
-0.0152, 0.0152, -0.0152, 0.0152,
0, -0.0006, -0.0000, 0.0006,
0.0006, 0, -0.0006, 0,
0.0106, 0.0106, 0.0106, 0.0106,
0, -1.4512, 0, 1.4512,
-1.4512, 0, 1.4512, 0,
```

```
-0.3049, 0.3049, -0.3049, 0.3049,
0, -0.0236, 0, 0.0236,
0.0236, 0, -0.0236, 0,
0.2107, 0.2107, 0.2107, 0.2107;

mpc::mat<Tny, Tnx> Cd;
Cd.setIdentity();

mpc::mat<Tny, Tnu> Dd;
Dd.setZero();

lmpc.setStateSpaceModel(Ad, Bd, Cd);

lmpc.setDisturbances(
mpc::mat<Tnx, Tndu>::Zero(),
mpc::mat<Tny, Tndu>::Zero());

mpc::cvec<Tnu> InputW, DeltaInputW;
mpc::cvec<Tny> OutputW;

OutputW << 0, 0, 10, 10, 10, 10, 0, 0, 0, 5, 5, 5;
InputW << 0.1, 0.1, 0.1, 0.1;
DeltaInputW << 0, 0, 0, 0;

lmpc.setObjectiveWeights(OutputW, InputW, DeltaInputW);

mpc::cvec<Tnx> xmin, xmax;
xmin << -M_PI / 6, -M_PI / 6, -mpc::inf, -mpc::inf, -mpc::inf,
    -1,
-mpc::inf, -mpc::inf, -mpc::inf, -mpc::inf, -mpc::inf, -mpc::
    inf;

xmax << M_PI / 6, M_PI / 6, mpc::inf, mpc::inf, mpc::inf, mpc::
    inf,
mpc::inf, mpc::inf, mpc::inf, mpc::inf, mpc::inf, mpc::inf;

mpc::cvec<Tny> ymin, ymax;
ymin.setOnes();
ymin *= -mpc::inf;
ymax.setOnes();
ymax *= mpc::inf;

mpc::cvec<Tnu> umin, umax;
double u0 = 10.5916;
umin << 9.6, 9.6, 9.6, 9.6;
umin.array() -= u0;
```

```
umax << 13, 13, 13, 13;
umax.array() -= u0;

lmpc.setConstraints(xmin, umin, ymin, xmax, umax, ymax);

mpc::cvec<Tny> yRef;
yRef << 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0;

lmpc.setReferences(yRef, mpc::cvec<Tnu>::Zero(), mpc::cvec<Tnu
    >::Zero());

auto res = lmpc.step(mpc::cvec<Tnx>::Zero(), mpc::cvec<Tnu>::
    Zero());
```

A.3.4 Example of non-linear MPC

This example shows how to drives the states of a Van der Pol oscillator to zero with constrained control input

```
double ts = 0.1;

nlmpc.setLoggerLevel(mpc::Logger::log_level::NORMAL);
nlmpc.setContinuosTimeModel(ts);

auto stateEq = [&](mpc::cvec<Tnx>& dx,
mpc::cvec<Tnx> x,
mpc::cvec<Tnu> u) {
        dx(0) = ((1.0 - (x(1) * x(1))) * x(0)) - x(1) + u(0);
        dx(1) = x(0);
};

nlmpc.setStateSpaceFunction(stateEq);

nlmpc.setObjectiveFunction([&](mpc::mat<Tph + 1, Tnx> x,
mpc::mat<Tph + 1, Tnu> u,
double) {
        return x.array().square().sum() + u.array().square().
            sum();
});

nlmpc.setIneqConFunction([&](mpc::cvec<ineq_c>& in_con,
mpc::mat<Tph + 1, Tnx>,
mpc::mat<Tph + 1, Tny>,
mpc::mat<Tph + 1, Tnu> u,
double) {
        for (int i = 0; i < ineq_c; i++) {
```

```
                in_con(i) = u(i, 0) - 0.5;
        }
});

mpc::cvec<Tnx> modelX, modeldX;

modelX(0) = 0;
modelX(1) = 1.0;

auto r = nlmpc.getLastResult();

for (;;) {
        r = nlmpc.step(modelX, r.cmd);
        stateEq(modeldX, modelX, r.cmd);
        modelX += modeldX * ts;

        if (std::fabs(modelX[0]) <= 1e-2 && std::fabs(modelX
            [1]) <= 1e-1) {
                break;
        }
}
```

Appendix B

Author's publications

This is the list of other projects in which I was involved during the PhD period that are not part of the main topic of this thesis.

N. Piccinelli, F. Vesentini, and R. Muradore, "Planning with real-time collision avoidance for cooperating agents under rigid body constraints," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, 2019, pp. 1261–1264

Abstract: In automated warehouses, path planning is a crucial topic to improve automation and efficiency. This kind of planning is usually computed off-line knowing the planimetry of the warehouse and the starting and target points of each agent. However, this global approach is not able to manage unexpected static/dynamic obstacles and other agents moving in the same area. For this reason in multi-robot systems global planners are usually integrated with local collision avoidance algorithms. In this paper we use the Voronoi diagram as global planner and the Velocity Obstacle (VO) method as collision avoidance algorithm. The goal of this paper is to extend such hybrid motion planner by enforcing mechanical constraints between agents in order to execute a task that cannot be performed by a single agent. We will focus on the cooperative task of carrying a payload, such as a bar. Two agents are constrained to move at the end points of the bar. We will improve the original algorithms by taking into account dynamically the constrained motion both at the global and at the collision avoidance level.

G. De Rossi, M. Minelli, A. Sozzi, N. Piccinelli, F. Ferraguti, F. Setti, M. Bonfé, C. Secchi, and R. Muradore, "Cognitive robotic architecture for semi-autonomous execution of manipulation tasks in a surgical environment," in

2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2019, pp. 7827–7833

Abstract: The development of robotic systems with a certain level of autonomy to be used in critical scenarios, such as an operating room, necessarily requires a seamless integration of multiple state-of-the-art technologies. In this paper we propose a cognitive robotic architecture that is able to help an operator accomplish a specific task. The architecture integrates an action recognition module to understand the scene, a supervisory control to make decisions, and a model predictive control to plan collision-free trajectory for the robotic arm taking into account obstacles and model uncertainty. The proposed approach has been validated on a simplified scenario involving only a da Vinci surgical robot and a novel manipulator holding standard laparoscopic tools.

C. Daffara, R. Muradore, N. Piccinelli, N. Gaburro, T. de Rubeis, and D. Ambrosini, "A cost-effective system for aerial 3d thermography of buildings," *Journal of Imaging*, vol. 6, no. 8, p. 76, 2020

Abstract: Three-dimensional (3D) imaging and infrared (IR) thermography are powerful tools in many areas in engineering and sciences. Their joint use is of great interest in the buildings sector, allowing inspection and non-destructive testing of elements as well as an evaluation of the energy efficiency. When dealing with large and complex structures, as buildings (particularly historical) generally are, 3D thermography inspection is enhanced by Unmanned Aerial Vehicles (UAV—also known as drones). The aim of this paper is to propose a simple and cost-effective system for aerial 3D thermography of buildings. Special attention is thus payed to instrument and reconstruction software choice. After a very brief introduction to IR thermography for buildings and 3D thermography, the system is described. Some experimental results are given to validate the proposal.

N. Sayols, A. Sozzi, N. Piccinelli, A. Hernansanz, A. Casals, M. Bonfè, and R. Muradore, "Global/local motion planning based on dynamic trajectory reconfiguration and dynamical systems for autonomous surgical robots," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 8483–8489

Abstract: This paper addresses the generation of collision-free trajectories

for the autonomous execution of assistive tasks in Robotic Minimally Invasive Surgery (R-MIS). The proposed approach takes into account geometric constraints related to the desired task, like for example the direction to approach the final target and the presence of moving obstacles. The developed motion planner is structured as a two-layer architecture: a global level computes smooth spline-based trajectories that are continuously updated using virtual potential fields; a local level, exploiting Dynamical Systems based obstacle avoidance, ensures collision free connections among the spline control points. The proposed architecture is validated in a realistic surgical scenario.

S. Aldegheri, N. Bombieri, F. Fummi, S. Girardi, R. Muradore, and N. Piccinelli, "Late breaking results: Enabling containerized computing and orchestration of ros-based robotic sw applications on cloud-server-edge architectures," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, IEEE, 2020, pp. 1–2

Abstract: We present a toolchain based on Docker and KubeEdge that enables containerization and orchestration of ROS-based robotic SW applications on heterogeneous and hierarchical HW architectures. The toolchain allows for verification of functional and real-time constraints through HW-in-the-loop simulation, and for automatic mapping exploration of the SW across Cloud-Server-Edge architectures. We present the results obtained for the deployment of a real case of study composed by an ORB-SLAM application combined to local/global planners with obstacle avoidance for a mobile robot navigation.

A. Roberti, N. Piccinelli, F. Falezza, G. De Rossi, S. Bonora, F. Setti, P. Fiorini, and R. Muradore, "A time-of-flight stereoscopic endoscope for anatomical 3d reconstruction," in *2021 International Symposium on Medical Robotics (ISMR)*, IEEE, 2021, pp. 1–7

Abstract: This paper presents a novel endoscope design for laparoscopic surgery that has been specifically tailored to provide both a stereoscopic view to the surgeon and a high-accuracy 3D reconstruction for an advanced visualization of the anatomical environment. The former helps the main surgeon in teleoperating a robotic minimally-invasive system (R-MIS) while the latter provides necessary data to upcoming autonomous surgical procedure implementations in a manner akin to the current development of autonomous driving systems. To this aim, we created an initial prototype that incorporates a pair of high-quality, chip-on-tip RGB cameras with a Time-of-Flight (ToF)

3D sensor in a sufficiently compact design to allow its usage in intra-luminal operations. The combination of these sensors provides a reliable 3D model of the anatomical structures at close and far distances within the workspace to effectively overcome the issues presented by current laparoscopy stereo endoscopes, for which the depth estimation is hindered by the reduced baseline distance between the cameras. Moreover, the application to current robotic platforms presents innate mathematical issues when applying hand-eye calibration techniques for localization. We finally developed a calibration procedure that merges both colour and depth information. The endoscope design is fully validated through the reconstruction of a planar surface, achieving a depth, latitudinal, and longitudinal orientation precision of 3.3mm, 0.02 rad, 0.025 rad respectively.

A. Pore, N. Piccinelli, G. De Rossi, M. Piano, D. Meli, D. Dall'Alba, P. Fiorini, and R. Muradore, "Endovine: Soft robotic endoscope for colonoscopy," in *The 14th Hamlyn Symposium on Medical Robotics*, 2022

Abstract: Colonoscopy is the gold standard examination procedure for screening colorectal cancer (CRC). Standard colonoscopy is carried out with a minimally invasive approach based on a flexible endoscope controlled by the steering knobs. However, for routine screening, the intrusive nature of the intervention, the rigidity of the device and the lack of intuitiveness in steering can make the experience highly unpleasant to the recipient. In this abstract, we present an Endoscopic Vine robot (EndoVine) designed for colonoscopy with a novel retraction mechanism. EndoVine comprises a double inverted soft inflatable sleeve that propels the tip using internal pneumatic propulsion. It will be equipped with an internal flexible channel for the camera, light source and working instruments to enable bowel inspection. The soft robotic body has been designed to be disposable, reducing the risk of infection transmission, which is a concern with traditional flexible endoscope.

Bibliography

[1] X. Du, M. Allan, A. Dore, S. Ourselin, D. Hawkes, J. D. Kelly, and D. Stoyanov, "Combined 2d and 3d tracking of surgical instruments for minimally invasive and robotic-assisted surgery," *International journal of computer assisted radiology and surgery*, vol. 11, no. 6, pp. 1109–1119, 2016.

[2] G. S. Guthart and K. J. Salisbury, "Intuitive telesurgery system: overview and application," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2000.

[3] U. Hagn, R. Konietschke, A. Tobergte, M. Nickl, S. Jörg, B. Kübler, G. Passig, M. Gröger, F. Fröhlich, U. Seibold, L. Le-Tien, A. Albu-Schäffer, A. Nothhelfer, F. Hacker, M. Grebenstein, and G. Hirzinger, "DLR MiroSurge: A versatile system for research in endoscopic telesurgery," *International Journal of Computer Assisted Radiology and Surgery*, 2010, ISSN: 18616429. DOI: 10.1007/s11548-009-0372-4.

[4] M. M. Marinho, B. V. Adorno, K. Harada, and M. Mitsuishi, "Dynamic active constraints for surgical robots using vector-field inequalities," *IEEE Transactions on Robotics*, vol. 35, no. 5, pp. 1166–1185, 2019.

[5] M. M. Marinho, B. V. Adorno, K. Harada, K. Deie, A. Deguet, P. Kazanzides, R. H. Taylor, and M. Mitsuishi, "A unified framework for the teleoperation of surgical robots in constrained workspaces," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 2721–2727.

[6] G.-Z. Yang, J. Cambias, K. Cleary, E. Daimler, J. Drake, P. E. Dupont, N. Hata, P. Kazanzides, S. Martel, R. V. Patel, *et al.*, "Medical robotics - regulatory, ethical, and legal considerations for increasing levels of autonomy," *Science Robotics*, vol. 2, no. 4, p. 8638, 2017.

[7] M. Ginesi, D. Meli, A. Roberti, N. Sansonetto, and P. Fiorini, "Autonomous task planning and situation awareness in robotic surgery*," in *2020 IEEE/RSJ International Conference on Intelligent Robots and*

Systems (IROS), 2020, pp. 3144–3150. DOI: 10.1109/IROS45743.2020.9341382.

[8] A. Sozzi, M. Bonfè, S. Farsoni, G. De Rossi, and R. Muradore, "Dynamic motion planning for autonomous assistive surgical robots," *Electronics*, vol. 8, no. 9, p. 957, 2019.

[9] G. De Rossi, M. Minelli, A. Sozzi, N. Piccinelli, F. Ferraguti, F. Setti, M. Bonfé, C. Secchi, and R. Muradore, "Cognitive robotic architecture for semi-autonomous execution of manipulation tasks in a surgical environment," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 7827–7833.

[10] P. Fiorini, D. Dall'Alba, M. Ginesi, B. Maris, D. Meli, H. Nakawala, A. Roberti, and E. Tagliabue, "Challenges of autonomous robotic surgery," in *Hamlyn Symposium on Medical Robotics (HSMR)*, 2019. DOI: 10.31256/HSMR2019.53.

[11] X. Ma, C. Song, P. W. Chiu, and Z. Li, "Autonomous flexible endoscope for minimally invasive surgery with enhanced safety," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2607–2613, 2019. DOI: 10.1109/LRA.2019.2895273.

[12] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[13] S. V. Raković and W. S. Levine, *Handbook of model predictive control*. Springer, 2018.

[14] B. Brogliato, R. Lozano, B. Maschke, and O. Egeland, "Dissipative systems analysis and control," *Theory and Applications*, vol. 2, 2007.

[15] A. Van Der Schaft, D. Jeltsema, *et al.*, "Port-hamiltonian systems theory: An introductory overview," *Foundations and Trends® in Systems and Control*, vol. 1, no. 2-3, pp. 173–378, 2014.

[16] V. R. Patel, K. K. Shah, R. K. Thaly, and H. Lavery, "Robotic-assisted laparoscopic radical prostatectomy: The ohio state university technique," *Journal of robotic surgery*, vol. 1, no. 1, pp. 51–59, 2007.

[17] E. Oleari, A. Leporini, D. Trojaniello, A. Sanna, U. Capitanio, F. Dehó, A. Larcher, F. Montorsi, A. Salonia, F. Setti, and R. Muradore, "Enhancing surgical process modeling for artificial intelligence development in robotics: The saras case study for minimally invasive procedures," in *International Symposium on Medical Information and Communica-*

tion Technology (ISMICT), IEEE, 2019. DOI: 10.1109/ISMICT.2019.8743931.

[18] F. Setti, E. Oleari, A. Leporini, D. Trojaniello, A. Sanna, U. Capitanio, F. Montorsi, A. Salonia, and R. Muradore, "A multirobots teleoperated platform for artificial intelligence training data collection in minimally invasive surgery," in *International Symposium on Medical Robotics (ISMR)*, IEEE, 2019.

[19] I. S. Gill, M. M. Desai, J. H. Kaouk, A. M. Meraney, D. P. Murphy, G. T. Sung, and A. C. Novick, "Laparoscopic partial nephrectomy for renal tumor: Duplicating open surgical techniques," *The Journal of urology*, vol. 167, no. 2 Part 1, pp. 469–476, 2002.

[20] F. Porpiglia, J. Renard, M. Billia, I. Morra, C. Scoffone, C. Cracco, R. Tarabuzzi, C. Terrone, and R. M. Scarpa, "Left laparoscopic radical nephrectomy with direct access to the renal artery: Technical advantages," *european urology*, vol. 49, no. 6, pp. 1004–1010, 2006.

[21] N. Piccinelli, E. Ghignoni, M. Piano, G. D. Rossi, F. Setti, and R. Muradore, "Prototyping a laparoscopic smart trocar for semi-autonomous surgical robotics," in *Conference on New Technologies for Computer and Robot Assisted Surgery (CRAS)*, 2022.

[22] C. D'Ettorre, A. Mariani, A. Stilli, P. Valdastri, A. Deguet, P. Kazanzides, R. H. Taylor, G. S. Fischer, S. P. DiMaio, A. Menciassi, *et al.*, "Accelerating surgical robotics research: Reviewing 10 years of research with the dvrk," *arXiv preprint arXiv:2104.09869*, 2021.

[23] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da vinci surgical system," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[24] J. Sandoval, G. Poisson, and P. Vieyres, "A new kinematic formulation of the rcm constraint for redundant torque-controlled robots," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2017, pp. 4576–4581.

[25] S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and M. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014, ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2014.01.005.

[26] A. Roberti, N. Piccinelli, D. Meli, R. Muradore, and P. Fiorini, "Improving rigid 3-d calibration for robotic surgery," *IEEE Transactions on Medical Robotics and Bionics*, vol. 2, no. 4, pp. 569–573, 2020.

[27] N. Piccinelli, A. Roberti, E. Tagliabue, F. Setti, G. Kronreif, R. Muradore, and P. Fiorini, "Rigid 3d registration of pre-operative information for semi-autonomous surgery," in *2020 International Symposium on Medical Robotics (ISMR)*, 2020, pp. 139–145. DOI: 10.1109/ISMR48331.2020.9312949.

[28] T. K. Adebar, A. E. Fletcher, and A. M. Okamura, "3-d ultrasound-guided robotic needle steering in biological tissue," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 12, pp. 2899–2910, 2014.

[29] M. Shah, R. D. Eastman, and T. Hong, "An overview of robot-sensor calibration methods for evaluation of perception systems," in *Proceedings of the Workshop on Performance Metrics for Intelligent Systems*, 2012, pp. 15–20.

[30] A. Tabb and K. M. Ahmad Yousef, "Solving the robot-world hand-eye (s) calibration problem with iterative methods," *Machine Vision and Applications*, vol. 28, no. 5, pp. 569–590, 2017.

[31] Z. Wang, Z. Liu, Q. Ma, A. Cheng, Y.-h. Liu, S. Kim, A. Deguet, A. Reiter, P. Kazanzides, and R. H. Taylor, "Vision-based calibration of dual rcm-based robot arms in human-robot collaborative minimally invasive surgery," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 672–679, 2017.

[32] K. Pachtrachai, M. Allan, V. Pawar, S. Hailes, and D. Stoyanov, "Hand-eye calibration for robotic assisted minimally invasive surgery without a calibration object," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2016, pp. 2485–2491.

[33] F. Zhong, Z. Wang, W. Chen, K. He, Y. Wang, and Y.-H. Liu, "Hand-eye calibration of surgical instrument for robotic surgery using interactive manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1540–1547, 2020.

[34] Y. C. Shiu and S. Ahmad, "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form ax= xb.," *IEEE Transactions on robotics and automation*, vol. 5, no. 1, pp. 16–29, 1989.

[35] J. C. Chou and M. Kamel, "Finding the position and orientation of a sensor on a robot manipulator using quaternions," *The international journal of robotics research*, vol. 10, no. 3, pp. 240–254, 1991.

[36] R.-h. Liang and J.-f. Mao, "Hand-eye calibration with a new linear decomposition algorithm," *Journal of Zhejiang University-SCIENCE A*, vol. 9, no. 10, pp. 1363–1368, 2008.

[37] H. Pan, N. L. Wang, and Y. S. Qin, "A closed-form solution to eye-to-hand calibration towards visual grasping," *Industrial Robot: An International Journal*, 2014.

[38] M. Najafi, N. Afsham, P. Abolmaesumi, and R. Rohling, "A closed-form differential formulation for ultrasound spatial calibration: Multi-wedge phantom," *Ultrasound in medicine & biology*, vol. 40, no. 9, pp. 2231–2243, 2014.

[39] S. Kahn, D. Haumann, and V. Willert, "Hand-eye calibration with a depth camera: 2d or 3d?" In *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, IEEE, vol. 3, 2014, pp. 481–489.

[40] D. W. Kim and J. E. Ha, "Hand/eye calibration using 3d-3d correspondences," in *Applied Mechanics and Materials*, Trans Tech Publ, vol. 319, 2013, pp. 532–535.

[41] S. Fuchs, "Calibration and multipath mitigation for increased accuracy of time-of-flight camera measurements in robotic applications," Ph.D. dissertation, Universitätsbibliothek der Technischen Universität Berlin, 2012.

[42] P. Mountney, D. Stoyanov, A. Davison, and G.-Z. Yang, "Simultaneous stereoscope localization and soft-tissue mapping for minimal invasive surgery," in *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2006.

[43] O. G. Grasa, J. Civera, and J. M. M. Montiel, "EKF monocular SLAM with relocalization for laparoscopic sequences," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[44] J. Totz, K. Fujii, P. Mountney, and G.-Z. Yang, "Enhanced visualisation for minimally invasive surgery," *International Journal of Computer Assisted Radiology and Surgery*, vol. 7, no. 3, pp. 423–432, 2012.

[45] J. Totz, P. Mountney, D. Stoyanov, and G.-Z. Yang, "Dense surface reconstruction for enhanced navigation in MIS," in *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2011.

[46] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[47] J. Song, J. Wang, L. Zhao, S. Huang, and G. Dissanayake, "Mis-slam: Real-time large-scale dense deformable slam system in minimal invasive surgery based on heterogeneous computing," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4068–4075, 2018.

[48] N. Mahmoud, A. Hostettler, T. Collins, L. Soler, C. Doignon, and J. Montiel, "SLAM based quasi dense reconstruction for minimally invasive surgery scenes," *arXiv preprint arXiv:1705.09107*, 2017.

[49] D. Burschka, M. Li, M. Ishii, R. H. Taylor, and G. D. Hager, "Scale-invariant registration of monocular endoscopic images to CT-scans for sinus surgery," *Medical Image Analysis*, vol. 9, no. 5, pp. 413–426, 2005.

[50] R. Muradore, P. Fiorini, G. Akgun, D. E. Barkana, M. Bonfe, F. Boriero, A. Caprara, G. D. Rossi, R. Dodi, O. J. Elle, F. Ferraguti, L. Gasperotti, R. Gassert, K. Mathiassen, D. Handini, O. Lambercy, L. Li, M. Kruusmaa, A. O. Manurung, G. Meruzzi, H. Q. P. Nguyen, N. Preda, G. Riolfo, A. Ristolainen, A. Sanna, C. Secchi, M. Torsello, and A. E. Yantac, "Development of a cognitive robotic system for simple surgical tasks," *International Journal of Advanced Robotic Systems*, vol. 12, no. 4, p. 37, 2015. DOI: 10.5772/60137.

[51] M. Bonfè, F. Boriero, R. Dodi, P. Fiorini, A. Morandi, R. Muradore, L. Pasquale, A. Sanna, and C. Secchi, "Towards automated surgical robotics: A requirements engineering approach," in *IEEE RAS EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 2012. DOI: 10.1109/BioRob.2012.6290700.

[52] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," in *Sensor fusion IV: control paradigms and data structures*, International Society for Optics and Photonics, vol. 1611, 1992, pp. 586–606.

[53] J. Yang, H. Li, and Y. Jia, "GO-ICP: Solving 3D registration efficiently and globally optimally," in *IEEE International Conference on Computer Vision (ICCV)*, 2013.

[54] Y. Sahillioğlu and L. Kavan, "Skuller: A volumetric shape registration algorithm for modeling skull deformities," *Medical image analysis*, vol. 23, no. 1, pp. 15–27, 2015.

[55] B. Lee and D. D. Lee, "Learning anisotropic ICP (LA-ICP) for robust and efficient 3D registration," in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2016.

[56] W. Gander and J. Hrebicek, *Solving problems in scientific computing using Maple and Matlab®*. Springer Science & Business Media, 2011.

[57] R. Y. Tsai, R. K. Lenz, *et al.*, "A new technique for fully autonomous and efficient 3 d robotics hand/eye calibration," *IEEE Transactions on robotics and automation*, vol. 5, no. 3, pp. 345–358, 1989.

[58] T. Haidegger, P. Kazanzides, I. Rudas, B. Benyó, and Z. Benyó, "The importance of accuracy measurement standards for computer-integrated interventional systems," in *EURON GEM Sig Workshop on The Role of Experiments in Robotics Research at IEEE ICRA*, 2010, pp. 1–6.

[59] D. M. Kwartowitz, S. D. Herrell, and R. L. Galloway, "Toward image-guided robotic surgery: Determining intrinsic accuracy of the da vinci robot," *International Journal of Computer Assisted Radiology and Surgery*, vol. 1, no. 3, pp. 157–165, 2006.

[60] M. O. Ward, G. Grinstein, and D. Keim, *Interactive Data Visualization: Foundations, Techniques, and Applications, Second Edition - 360 Degree Business*, 2nd. USA: A. K. Peters, Ltd., 2015, ISBN: 1482257378.

[61] P. A. Yushkevich, J. Piven, H. Cody Hazlett, R. Gimpel Smith, S. Ho, J. C. Gee, and G. Gerig, "User-guided 3D active contour segmentation of anatomical structures: Significantly improved efficiency and reliability," *Neuroimage*, vol. 31, no. 3, pp. 1116–1128, 2006.

[62] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Eurographics Symposium on Geometry Processing (SGP)*, 2006.

[63] H. C. Longuet-Higgins, "The reconstruction of a plane surface from two perspective projections," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 227, no. 1249, pp. 399–410, 1986.

[64] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[65] Y. Zhong, "Intrinsic shape signatures: A shape descriptor for 3d object recognition," in *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2009. DOI: 10.1109/ICCVW.2009.5457637.

[66] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *2009 IEEE international conference on robotics and automation*, IEEE, 2009, pp. 3212–3217.

[67] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *IEEE International Conference on Robotics and Automation (ICRA)*, 1991. DOI: 10.1109/ROBOT.1991.132043.

[68] F. Falezza, N. Piccinelli, G. De Rossi, A. Roberti, G. Kronreif, F. Setti, P. Fiorini, and R. Muradore, "Modeling of surgical procedures using statecharts for semi-autonomous robotic surgery," *IEEE Transactions on Medical Robotics and Bionics*, vol. 3, no. 4, pp. 888–899, 2021.

[69] A. Rector and J. Rogers, "Ontological and practical issues in using a description logic to represent medical concept systems: Experience from galen," in *Reasoning Web International Summer School*, Springer, 2006, pp. 197–231.

[70] T. Haidegger, "Autonomy for surgical robots: Concepts and paradigms," *IEEE Transactions on Medical Robotics and Bionics*, vol. 1, no. 2, pp. 65–76, 2019. DOI: 10.1109/TMRB.2019.2913282.

[71] L. Longo, "Argumentation for knowledge representation, conflict resolution, defeasible inference and its integration with machine learning," in *Machine Learning for Health Informatics*, Springer, 2016, pp. 183–208.

[72] D. Harel and A. Pnueli, "On the development of reactive systems," in *Logics and models of concurrent systems*, Springer, 1985, pp. 477–498.

[73] S. Van Mierlo and H. Vangheluwe, "Statecharts: A formalism to model, simulate and synthesize reactive and autonomous timed systems," in *Foundations of Multi-Paradigm Modelling for Cyber-Physical Systems*, Springer, 2020, pp. 155–176.

[74] Q. Zhao and B. H. Krogh, "Formal verification of statecharts using finite-state model checkers," *IEEE transactions on control systems technology*, vol. 14, no. 5, pp. 943–950, 2006.

[75] B. Sobolev, D. Harel, C. Vasilakis, and A. Levy, "Using the statecharts paradigm for simulation of patient flow in surgical care," *Health Care Management Science*, vol. 11, no. 1, pp. 79–86, 2008.

[76] F. Lalys and P. Jannin, "Surgical process modelling: A review," *International journal of computer assisted radiology and surgery*, vol. 9, no. 3, pp. 495–511, 2014.

[77] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, N. SINGH, and J. Schneider, "Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Mar. 2020.

[78] G. Forestier, F. Petitjean, P. Senin, L. Riffaud, P. L. Henaux, and P. Jannin, "Finding discriminative and interpretable patterns in sequences of surgical activities," *Artificial Intelligence in Medicine*, vol. 82, pp. 11–19, 2017, ISSN: 18732860. DOI: 10.1016/j.artmed.2017.09.002.

[79] O. Dergachyova, D. Bouget, A. Huaulmé, X. Morandi, and P. Jannin, "Automatic data-driven real-time segmentation and recognition of surgical workflow," *International Journal of Computer Assisted Radiology and Surgery*, vol. 11, no. 6, pp. 1081–1089, 2016, ISSN: 18616429. DOI: 10.1007/s11548-016-1371-x.

[80] A. Murali, S. Sen, B. Kehoe, A. Garg, S. McFarland, S. Patil, W. D. Boyd, S. Lim, P. Abbeel, and K. Goldberg, "Learning by observation for surgical subtasks: Multilateral cutting of 3D viscoelastic and 2D Orthotropic Tissue Phantoms," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June, pp. 1202–1209, 2015, ISSN: 10504729. DOI: 10.1109/ICRA.2015.7139344.

[81] S. Sen, A. Garg, D. V. Gealy, S. McKinley, Y. Jen, and K. Goldberg, "Automating multi-throw multilateral surgical suturing with a mechanical needle guide and sequential convex optimization," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4178–4185. DOI: 10.1109/ICRA.2016.7487611.

[82] T. D. Nagy and T. Haidegger, "A dvrk-based framework for surgical subtask automation," *Acta Polytechnica Hungarica*, pp. 61–78, 2019.

[83] M. Hwang, D. Seita, B. Thananjeyan, J. Ichnowski, S. Paradis, D. Fer, T. Low, and K. Goldberg, "Applying depth-sensing to automated surgical manipulation with a da vinci robot," in *2020 International Symposium on Medical Robotics (ISMR)*, 2020, pp. 22–29. DOI: 10.1109/ISMR48331.2020.9312948.

[84] A. Kurt and Ü. Özgüner, "Hierarchical finite state machines for autonomous mobile systems," *Control Engineering Practice*, vol. 21, no. 2, pp. 184–194, 2013.

[85] M. Colledanchise and P. Ögren, "How behavior trees modularize hybrid control systems and generalize sequential behavior compositions, the subsumption architecture, and decision trees," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 372–389, 2017. DOI: 10.1109/TRO.2016.2633567.

[86] D. Hu, Y. Gong, B. Hannaford, and E. J. Seibel, "Semi-autonomous simulated brain tumor ablation with ravenii surgical robot using behavior tree," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 3868–3875. DOI: 10.1109/ICRA.2015.7139738.

[87] M. Ginesi, D. Meli, H. Nakawala, A. Roberti, and P. Fiorini, "A knowledge-based framework for task automation in surgery," in *2019 19th International Conference on Advanced Robotics (ICAR)*, IEEE, 2019, pp. 37–42.

[88] Z. Dogmus, G. Gezici, V. Patoglu, and E. Erdem, "Developing and maintaining an ontology for rehabilitation robotics.," in *KEOD*, 2012, pp. 389–395.

[89] C. E. Reiley and G. D. Hager, "Task versus subtask surgical skill evaluation of robotic minimally invasive surgery," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5761 LNCS, no. PART 1, pp. 435–442, 2009, ISSN: 03029743. DOI: 10.1007/978-3-642-04268-3_54.

[90] M. Ghallab, D. Nau, and P. Traverso, *Automated Planning: theory and practice*. Elsevier, 2004.

[91] C. E. Reiley, H. C. Lin, B. Varadarajan, B. Vagvolgyi, S. Khudanpur, D. D. Yuh, and G. D. Hager, "Automatic recognition of surgical motions using statistical modeling for capturing variability," in *MMVR*, vol. 132, 2008, pp. 396–401.

[92] T. Bylander, "The computational complexity of propositional strips planning," *Artificial Intelligence*, vol. 69, no. 1-2, pp. 165–204, 1994.

[93] J. Shi *et al.*, "Good features to track," in *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, IEEE, 1994, pp. 593–600.

[94] J.-Y. Bouguet *et al.*, "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm," *Intel corporation*, vol. 5, no. 1-10, p. 4, 2001.

[95] G. Welch, G. Bishop, *et al.*, "An introduction to the kalman filter," 1995.

[96] M. Minelli, N. Piccinelli, F. Falezza, F. Ferraguti, R. Muradore, and C. Secchi, *IEEE Transactions on Control Systems Technology*, 2022, Accepted.

[97] F. Loschi, N. Piccinelli, D. Dall'Alba, R. Muradore, P. Fiorini, and C. Secchi, "An optimized two-layer approach for efficient and robustly stable bilateral teleoperation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 12 449–12 455.

[98] B. Bethea, A. Okamura, M. Kitagawa, T. Fitton, S. Cattaneo, M. Ameli, W. Baumgartner, and D. Yuh, "Application of haptic feedback to robotic surgery," *Journal of laparoendoscopic & advanced surgical techniques. Part A*, vol. 14, no. 3, pp. 191–5, Jul. 2004. DOI: 10.1089/1092642041255441.

[99] M. Franken, S. Stramigioli, S. Misra, C. Secchi, and A. MacChelli, "Bilateral telemanipulation with time delays: A two-layer approach combining passivity and transparency," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 741–756, 2011, ISSN: 15523098. DOI: 10.1109/TRO.2011.2142430.

[100] F. Ferraguti, N. Preda, A. Manurung, M. Bonfe, O. Lambercy, R. Gassert, R. Muradore, P. Fiorini, and C. Secchi, "An energy tank-based interactive control architecture for autonomous and teleoperated robotic surgery," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1073–1088, 2015.

[101] S. Stramigioli, C. Secchi, A. J. van der Schaft, and C. Fantuzzi, "Sampled data systems passivity and discrete port-hamiltonian systems," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 574–587, 2005.

[102] D. A. Lawrence, "Stability and transparency in bilateral teleoperation," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 5, pp. 624–637, Oct. 1993, ISSN: 1042-296X. DOI: 10.1109/70.258054.

[103] J. H. Ryu, J. Artigas, and C. Preusche, "A passive bilateral control scheme for a teleoperator with time-varying communication delay," *Mechatronics*, vol. 20, no. 7, pp. 812–823, 2010, ISSN: 09574158. DOI: 10.1016/j.mechatronics.2010.07.006.

[104] G. Niemeyer and J.-J. E. Slotine, "Telemanipulation with time delays," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 873–890, 2004.

[105] R. J. Anderson and M. W. Spong, "Bilateral control of teleoperators with time delay," in *Proceedings of the 1988 IEEE International Conference on Systems, Man, and Cybernetics*, IEEE, vol. 1, 1988, pp. 131–138.

[106] G. Niemeyer and J. J. E. Slotine, "Stable Adaptive Teleoperation," *IEEE Journal of Oceanic Engineering*, vol. 16, no. 1, pp. 152–162, 1991, ISSN: 15581691. DOI: 10.1109/48.64895.

[107] D. Lee and M. W. Spong, "Passive bilateral teleoperation with constant time delay," *IEEE transactions on robotics*, vol. 22, no. 2, pp. 269–281, 2006.

[108] N. Chopra, M. W. Spong, and R. Lozano, "Synchronization of bilateral teleoperators with time delay," *Automatica*, vol. 44, no. 8, pp. 2142–2148, 2008.

[109] B. Hannaford and J.-H. Ryu, "Time-domain passivity control of haptic interfaces," *IEEE transactions on Robotics and Automation*, vol. 18, no. 1, pp. 1–10, 2002.

[110] J.-H. Ryu, D.-S. Kwon, and B. Hannaford, "Stable teleoperation with time-domain passivity control," *IEEE Transactions on robotics and automation*, vol. 20, no. 2, pp. 365–373, 2004.

[111] Y. Ye, Y.-J. Pan, and Y. Gupta, "A power based time domain passivity control for haptic interfaces," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, IEEE, 2009, pp. 7521–7526.

[112] O. J. Smith, "Closed control of loop with dead time," *Chemical engineering progress*, vol. 53, pp. 217–219, 1957.

[113] J.-Q. Huang and F. L. Lewis, "Neural-network predictive control for nonlinear dynamic systems with time-delay," *IEEE Transactions on Neural Networks*, vol. 14, no. 2, pp. 377–389, 2003.

[114] K. Fite, M. Goldfarb, and A. Rubio, "Transparent telemanipulation in the presence of time delay," in *Proceedings 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, IEEE, vol. 1, 2003, pp. 254–259.

[115] A. C. Smith and K. Van Hashtrudi-Zaad, "Neural network-based teleoperation using smith predictors," in *IEEE International Conference Mechatronics and Automation, 2005*, IEEE, vol. 3, 2005, pp. 1654–1659.

[116] J. Artigas, J.-H. Ryu, and C. Preusche, "Time domain passivity control for position-position teleoperation architectures," *Presence: Teleoperators and Virtual Environments*, vol. 19, no. 5, pp. 482–497, 2010.

[117] D. Lee and K. Huang, "Passive-set-position-modulation framework for interactive robotic systems," *IEEE Transactions on Robotics*, vol. 26, no. 2, pp. 354–369, 2010, ISSN: 15523098. DOI: 10.1109/TRO.2010.2041877.

[118] M. Minelli, F. Ferraguti, N. Piccinelli, R. Muradore, and C. Secchi, "An energy-shared two-layer approach for multi-master-multi-slave bilateral teleoperation systems," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 423–429. DOI: 10.1109/ICRA.2019.8794335.

[119] F. Ferraguti, C. Secchi, and C. Fantuzzi, "A tank-based approach to impedance control with variable stiffness," in *2013 IEEE International Conference on Robotics and Automation*, IEEE, 2013, pp. 4948–4953.

[120] A. Franchi, C. Secchi, H. I. Son, H. H. Bulthoff, and P. R. Giordano, "Bilateral teleoperation of groups of mobile robots with time-varying topology," *Ieee transactions on robotics*, vol. 28, no. 5, pp. 1019–1033, 2012.

[121] V. Duindam and S. Stramigioli, "Port-based asymptotic curve tracking for mechanical systems," *European Journal of Control*, vol. 10, no. 5, pp. 411–420, 2004, ISSN: 0947-3580. DOI: https://doi.org/10.3166/ejc.10.411-420.

[122] C. Secchi and F. Ferraguti, "Energy optimization for a robust and flexible interaction control," in *2019 International Conference on Robotics and Automation (ICRA)*, Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019, pp. 1919–1925.

[123] F. Ferraguti, M. Bonfe, C. Fantuzzi, and C. Secchi, "Optimized power modulation in wave based bilateral teleoperation," *IEEE/ASME Transactions on Mechatronics*, pp. 1–1, 2020.

[124] F. Ferraguti, N. Preda, G. De Rossi, M. Bonfe, R. Muradore, P. Fiorini, and C. Secchi, "A two-layer approach for shared control in semi-autonomous robotic surgery," in *2015 European Control Conference, ECC 2015*, 2015, ISBN: 9783952426937. DOI: 10.1109/ECC.2015.7330632.

[125] C. Secchi, S. Stramigioli, and C. Fantuzzi, *Control of Interactive Robotic Interfaces: a port-Hamiltonian Approach*. Springer, 2007, ISBN: 978-3540497127.

[126] B. Hannaford, L. Wood, D. A. McAffee, and H. Zak, "Performance evaluation of a six-axis generalized force-reflecting teleoperator," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, no. 3, pp. 620–633, 1991.

[127] N. Piccinelli and R. Muradore, "A passivity-based bilateral teleoperation architecture using distributed nonlinear model predictive control," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2020, pp. 11 466–11 472.

[128] N. Piccinelli and R. Muradore, "A bilateral teleoperation with interaction force constraint in unknown environment using non linear model predictive control," *European Journal of Control*, vol. 62, pp. 185–191, 2021.

[129] N. Piccinelli and R. Muradore, "Interaction force constraints for position-based manipulator using linear mpc," Under review.

[130] N. Piccinelli and R. Muradore, "Control of interaction force using hybrid linear model predictive control," in *2022 Mediterranean Conference on Control and Automation (MED)*, 2022.

[131] N. Piccinelli and R. Muradore, "Passivity-based bilateral teleoperation using linear model predictive force control," Under review.

[132] A. Bemporad, "Predictive control of teleoperated constrained systems with unbounded communication delays," in *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No.98CH36171)*, vol. 2, Dec. 1998, 2133–2138 vol.2. DOI: 10.1109/CDC.1998.758651.

[133] S. Sirouspour and A. Shahdi, "Model predictive control for transparent teleoperation under communication time delay," *IEEE Transactions on Robotics*, 2006, ISSN: 15523098. DOI: 10.1109/TRO.2006.882939.

[134] J. Sheng and M. W. Spong, "Model predictive control for bilateral teleoperation systems with time delays," in *Canadian Conference on Electrical and Computer Engineering 2004 (IEEE Cat. No.04CH37513)*, vol. 4, May 2004, 1877–1880 Vol.4. DOI: 10.1109/CCECE.2004.1347575.

[135] T. Slama, A. Trevisani, D. Aubry, R. Oboe, and F. Kratz, "Experimental analysis of an internet-based bilateral teleoperation system with motion and force scaling using a model predictive controller," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 9, pp. 3290–3299, Sep. 2008. DOI: 10.1109/TIE.2008.928120.

[136] L. Santos and R. Cortesão, "Computed-torque control for robotic-assisted tele-echography based on perceived stiffness estimation," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 3, pp. 1337–1354, 2018.

[137] J. Park and O. Khatib, "A haptic teleoperation approach based on contact force control," *The International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 575–591, 2006. DOI: 10.1177/0278364906065385.

[138] M. Bednarczyk, H. Omran, and B. Bayle, "Model predictive impedance control," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 4702–4708. DOI: 10.1109/ICRA40945.2020.9196969.

[139] A. Wahrburg and K. Listmann, "Mpc-based admittance control for robotic manipulators," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 7548–7554. DOI: 10.1109/CDC.2016.7799435.

[140] T. Gold, A. Völz, and K. Graichen, "Model predictive position and force trajectory tracking control for robot-environment interaction," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 7397–7402. DOI: 10.1109/IROS45743.2020.9341168.

[141] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[142] J. K. Mills and D. M. Lokhorst, "Stability and control of robotic manipulators during contact/noncontact task transition," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 3, pp. 335–345, 1993.

[143] D. Heck, A. Saccon, N. Van de Wouw, and H. Nijmeijer, "Guaranteeing stable tracking of hybrid position–force trajectories for a robot manipulator interacting with a stiff environment," *Automatica*, vol. 63, pp. 235–247, 2016.

[144] P. R. Pagilla and B. Yu, "A stable transition controller for constrained robots," *IEEE/ASME transactions on mechatronics*, vol. 6, no. 1, pp. 65–74, 2001.

[145] C. Y. Lai, Y. P. Li, N. D. Vuong, T. M. Lim, C. Y. Ma, and C. W. Lim, "Nonlinear damping for improved transient performance in robotics force control," in *2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, IEEE, 2012, pp. 314–319.

[146] Z. Doulgeri and G. Iliadis, "Contact stability analysis of a one degree-of-freedom robot using hybrid system stability theory," *Robotica*, vol. 23, no. 5, pp. 607–614, 2005.

[147] T. Schlegl, M. Buss, and G. Schmidt, "A hybrid systems approach toward modeling and dynamical simulation of dextrous manipulation," *IEEE/ASME transactions on mechatronics*, vol. 8, no. 3, pp. 352–361, 2003.

[148] R. Carloni, R. G. Sanfelice, A. R. Teel, and C. Melchiorri, "A hybrid control strategy for robust contact detection and force regulation," in *2007 American Control Conference*, IEEE, 2007, pp. 1461–1466.

[149] M. Martino and M. E. Broucke, "A reach control approach to bumpless transfer of robotic manipulators," in *53rd IEEE Conference on Decision and Control*, IEEE, 2014, pp. 25–30.

[150] D. Nicolis, A. M. Zanchettin, and P. Rocco, "Constraint-based and sensorless force control with an application to a lightweight dual-arm robot," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 340–347, 2016.

[151] S. S. M. Salehian and A. Billard, "A dynamical-system-based approach for controlling robotic manipulators during noncontact/contact transitions," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2738–2745, 2018.

[152] K. Kitagaki and M. Uchiyama, "Optimal approach velocity of an end-effector to the environment," *Advanced Robotics*, vol. 8, no. 2, pp. 123–137, 1993.

[153] K. Alexis, C. Huerzeler, and R. Siegwart, "Hybrid modeling and control of a coaxial unmanned rotorcraft interacting with its environment through contact," in *2013 IEEE International Conference on Robotics and Automation*, IEEE, 2013, pp. 5417–5424.

[154] A. Albu-Schaffer and G. Hirzinger, "Cartesian impedance control techniques for torque controlled light-weight robots," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, vol. 1, 2002, 657–663 vol.1. DOI: 10.1109/ROBOT.2002.1013433.

[155] A. Ajoudani, A. M. Zanchettin, S. Ivaldi, A. Albu-Schäffer, K. Kosuge, and O. Khatib, "Progress and prospects of the human–robot collaboration," *Autonomous Robots*, vol. 42, no. 5, pp. 957–975, 2018.

[156] W. Amanhoud, M. Khoramshahi, M. Bonnesoeur, and A. Billard, "Force adaptation in contact tasks with dynamical systems," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 6841–6847.

[157] F. Ficuciello, L. Villani, and B. Siciliano, "Variable impedance control of redundant manipulators for intuitive human–robot physical interaction," *IEEE Transactions on Robotics*, vol. 31, no. 4, pp. 850–863, 2015.

[158] W. He, C. Xue, X. Yu, Z. Li, and C. Yang, "Admittance-based controller design for physical human–robot interaction in the constrained task space," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 4, pp. 1937–1949, 2020.

[159] K. J. Kazim, J. Bethge, J. Matschek, and R. Findeisen, "Combined predictive path following and admittance control," in *2018 Annual American Control Conference (ACC)*, IEEE, 2018, pp. 3153–3158.

[160] J. Matschek, J. Bethge, P. Zometa, and R. Findeisen, "Force feedback and path following using predictive control: Concept and application to a lightweight robot," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 9827–9832, 2017, 20th IFAC World Congress, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2017.08.898.

[161] D. Müller, A. Mayer, and O. Sawodny, "Model predictive force control for robots in compliant environments with guaranteed maximum force," in *2019 American Control Conference (ACC)*, 2019, pp. 1355–1360. DOI: 10.23919/ACC.2019.8814664.

[162] J. de la Casa Cárdenas, A. S. García, S. S. Martínez, J. G. García, and J. G. Ortega, "Model predictive position/force control of an anthropomorphic robotic arm," in *2015 IEEE International Conference on Industrial Technology (ICIT)*, 2015, pp. 326–331. DOI: 10.1109/ICIT.2015.7125119.

[163] J. Matschek, R. Jordanowa, and R. Findeisen, "Direct force feedback using gaussian process based model predictive control," in *2020 IEEE Conference on Control Technology and Applications (CCTA)*, 2020, pp. 8–13. DOI: 10.1109/CCTA41146.2020.9206279.

[164] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*, 1st. Springer Publishing Company, Incorporated, 2008, ISBN: 1846286417.

[165] G. Gilardi and I. Sharf, "Literature survey of contact dynamics modelling," *Mechanism and machine theory*, vol. 37, no. 10, pp. 1213–1239, 2002.

[166] A. Wahrburg, E. Morara, G. Cesari, B. Matthias, and H. Ding, "Cartesian contact force estimation for robotic manipulators using kalman filters and the generalized momentum," in *IEEE International Conference on Automation Science and Engineering*, 2015. DOI: 10.1109/CoASE.2015.7294266.

[167] G. Garofalo, N. Mansfeld, J. Jankowski, and C. Ott, "Sliding mode momentum observers for estimation of external torques and joint acceleration," in *International Conference on Robotics and Automation*, 2019, pp. 6117–6123.

[168] A. De Luca, A. Albu-Schaffer, S. Haddadin, and G. Hirzinger, "Collision detection and safe reaction with the dlr-iii lightweight manipulator arm," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2006, pp. 1623–1630.

[169] R. Muradore, R. Foroncelli, and P. Fiorini, "Statistical methods for estimating the dynamical parameters of manipulators," in *IEEE Conference on Decision and Control (CDC) held jointly with 28th Chinese Control Conference*, 2009, pp. 6472–6477. DOI: 10.1109/CDC.2009.5400194.

[170] M. W. Spong, S. Hutchinson, M. Vidyasagar, *et al.*, *Robot modeling and control*. 2006.

[171] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[172] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020. DOI: 10.1007/s12532-020-00179-2.

[173] P. Falugi, "Model predictive control: A passive scheme," *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 19, no. 3, pp. 1017–1022, 2014, ISSN: 14746670. DOI: 10.3182/20140824-6-za-1003.02165.

[174] J. P. Koeln and A. G. Alleyne, "Stability of decentralized model predictive control of graph-based power flow systems via passivity," *Automatica*, vol. 82, pp. 29–34, 2017.

[175] A. Bemporad, "Efficient conversion of mixed logical dynamical systems into an equivalent piecewise affine form," *IEEE transactions on Automatic Control*, vol. 49, no. 5, pp. 832–838, 2004.

[176] A. Bemporad, *Hybrid Toolbox - User's Guide*, http://cse.lab.imtlucca.it/~bemporad/hybrid/toolbox, 2004.

[177] M. Kvasnica, P. Grieder, M. Baotić, and M. Morari, "Multi-parametric toolbox (mpt)," in *International workshop on hybrid systems: Computation and control*, Springer, 2004, pp. 448–462.

[178] S. G. Johnson, *The nlopt nonlinear-optimization package*, http://github.com/stevengj/nlopt.

[179] N. Piccinelli, F. Vesentini, and R. Muradore, "Planning with real-time collision avoidance for cooperating agents under rigid body constraints," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, 2019, pp. 1261–1264.

[180] G. De Rossi, M. Minelli, A. Sozzi, N. Piccinelli, F. Ferraguti, F. Setti, M. Bonfé, C. Secchi, and R. Muradore, "Cognitive robotic architecture for semi-autonomous execution of manipulation tasks in a surgical environment," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 7827–7833.

[181] C. Daffara, R. Muradore, N. Piccinelli, N. Gaburro, T. de Rubeis, and D. Ambrosini, "A cost-effective system for aerial 3d thermography of buildings," *Journal of Imaging*, vol. 6, no. 8, p. 76, 2020.

[182] N. Sayols, A. Sozzi, N. Piccinelli, A. Hernansanz, A. Casals, M. Bonfè, and R. Muradore, "Global/local motion planning based on dynamic trajectory reconfiguration and dynamical systems for autonomous surgical robots," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 8483–8489.

[183] S. Aldegheri, N. Bombieri, F. Fummi, S. Girardi, R. Muradore, and N. Piccinelli, "Late breaking results: Enabling containerized computing and orchestration of ros-based robotic sw applications on cloud-server-edge architectures," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, IEEE, 2020, pp. 1–2.

[184] A. Roberti, N. Piccinelli, F. Falezza, G. De Rossi, S. Bonora, F. Setti, P. Fiorini, and R. Muradore, "A time-of-flight stereoscopic endoscope for anatomical 3d reconstruction," in *2021 International Symposium on Medical Robotics (ISMR)*, IEEE, 2021, pp. 1–7.

[185] A. Pore, N. Piccinelli, G. De Rossi, M. Piano, D. Meli, D. Dall'Alba, P. Fiorini, and R. Muradore, "Endovine: Soft robotic endoscope for colonoscopy," in *The 14th Hamlyn Symposium on Medical Robotics*, 2022.