2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Integrating Model Predictive Control and Dynamic Waypoints Generation for Motion Planning in Surgical Scenario

Marco Minelli[1], Alessio Sozzi[2], Giacomo De Rossi[3], Federica Ferraguti[1], Francesco Setti[3], Riccardo Muradore[3], Marcello Bonfè[2], Cristian Secchi[1]

Abstract— **In this paper we present a novel strategy for motion planning of autonomous robotic arms in Robotic Minimally Invasive Surgery (R-MIS). We consider a scenario where several laparoscopic tools must move and coordinate in a shared environment. The motion planner is based on a Model Predictive Controller (MPC) that predicts the future behavior of the robots and allows to move them avoiding collisions between the tools and satisfying the velocity limitations. In order to avoid the local minima that could affect the MPC, we propose a strategy for driving it through a sequence of waypoints. The proposed control strategy is validated on a realistic surgical scenario.**

I. INTRODUCTION

Autonomous robots in surgical practice are not common, even though robotic surgery is reality since when Intuitive Surgical released the da Vinci®system almost 20 years ago. Introducing autonomy in medical robotics is challenging because of both ethical and technical issues. As for the ethical aspects, one should consider if human safety can be more at risk, while technically the execution of surgical tasks is hardly predictable, in particular when soft body parts and tissues are involved [1]–[4]. Nevertheless, the research on autonomous surgical robots is quite active and the medical community is looking forward for new results. For example, [5] proposed a cognitive control architecture designed to operate a surgical robot for needle insertion and suturing tasks in either teleoperated [6] and autonomous mode [7], guaranteeing a stable switch between the two and an adaptive interaction with the environment in both modes [8]. Automated suturing is investigated by means of advanced learning techniques [9], [10] or offline geometry-based motion planning techniques [11], [12]. A survey on robotized needle insertion is in [13].

In Robotic Minimally Invasive Surgery (R-MIS) the surgical tools are inserted into the abdomen of the patient through *trocars*, that impose a Remote Center of Motion (RCM) to the surgical tools. Developing autonomous robots for R-MIS operations is further complicated, not only because of the motion constraint given by trocars, but also because such operations require the cooperation of several human actors. In particular, in R-MIS the main surgeon is seated at the da Vinci®console and remotely controls its tools, while an assistant surgeon handles standard laparoscopic tools to perform secondary tasks (*e.g.* move organs or suction blood). A recent EU funded H2020 research project SARAS[1] (*Smart Autonomous Robotic Assistant Surgeon*) has the goal to substitute the assistant surgeon with an autonomous robotic system. To this aim, we developed a cognitive system that mimics the behavior of the assistant surgeon. This cognitive system is trained on data acquired with the SARAS Multi-robots Surgery Platform [14]–[16] when the assistant surgeon is teleoperating assistive robots [17]. In parallel, we are developing robot motion planners taking into account dynamic environments, characterized by the presence of obstacles (*e.g.* tools and patient's organs) whose motion is predictable only within a short time horizon. This aspect encourages the use of reactive methods like [18], and discourages the use of geometry-based and sampling-based planning algorithms.

In this paper we propose a reactive approach to motion planning for a multi-arms laparoscopic surgical robot in a dynamic environment, based on Model Predictive Control (MPC). MPC-based approaches are a good choice in our scenario because they rapidly converge towards optimal solutions and allow to account for different types of constraints such as velocity limits and distance to obstacles [19]. Nevertheless, embedding the kinematic constraints of R-MIS in a MPC problem can be cumbersome. In this paper we will show how to complement the MPC-based approach with a waypoint computation strategy that guides the obstacle avoidance manoeuvre towards favourable directions and reduces the risk of being trapped into local minima.

The main contributions of this paper are:

- A Model Predictive Controller (MPC) that computes the collision-free trajectory of multiple laparoscopic surgical tools to their target positions, considering the tools as a whole.
- A novel strategy to compute suitable waypoints in the workspace, improving the convergence of the motions generated by the MPC towards the target positions, while avoiding obstacles.
- An experimental validation of the overall control architecture in a realistic surgical scenario.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 779813 (SARAS).

[1] M. Minelli, F. Ferraguti and C. Secchi are with the Dept. of Sciences and Methods of Engineering, University of Modena and Reggio Emilia, Italy. marco.minelli@unimore.it
[2] A. Sozzi and M. Bonfè are with the Dept. of Engineering, University of Ferrara, Italy.
[3] G. De Rossi, F. Setti and R. Muradore are with the Dept. of Computer Science, University of Verona, Italy.

[1]saras-project.eu

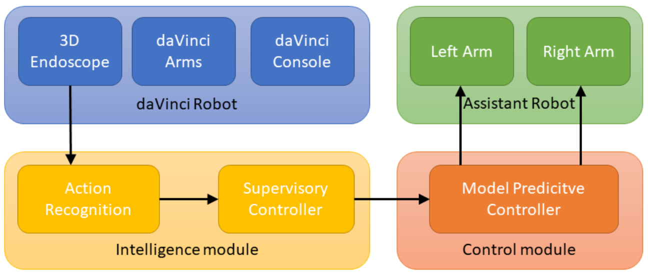


Fig. 1. Control architecture scheme for robotic laparoscopic surgery.

II. CONTROL ARCHITECTURE FOR ROBOTIC LAPAROSCOPIC SURGERY

We build on the control architecture for R-MIS proposed in [16] and summarized in Fig. 1. The overall architecture can be split in two modules: 1) the *Intelligence module*, that determines the desired behavior of the system, and 2) the *Control module*, that controls the robots in order to implement the desired behavior. In particular, the *Action Recognition* sub-module exploits the endoscopic images to detect actions executed during the surgical procedure. The most likely action is then provided to the *Supervisory Control* sub-model, together with a confidence level associated with that detection. The progress of the surgical procedure is monitored by the *Supervisory Control* that computes the targets where the robots have to move, and sends them to the *Control module*, together with the confidence level. A *Model Predictive Controller (MPC)* is used to translate such targets into feasible trajectories for the arms.

In this paper we will focus on the *Control module*. We will consider the SARAS Multirobots Surgery Platform [14], where four laparoscopic tools operate in a shared environment. However, the approach presented in this paper is general and can be applied to any system for laparoscopic surgery where tools guided by robotic arms are used. In the SARAS setup, two laparoscopic tools, namely the *controlled tools*, are controlled by the autonomous system (Assistant robot left and right arms in Fig. 1), the remaining two arms, teleoperated by the main surgeon, are located in the workspace and considered as *obstacles*. The workspace of interest for collision-avoidance is the internal part of the patient's abdomen; the control system moves autonomously the controlled tools in order to reach the target configurations while dynamically avoiding collisions with the obstacles and between the two controlled tools. The main control system is based on an MPC where the mathematical model of the two controlled tools is used to predict their future behavior and to compute the optimal input with which the robots can be controlled. Based on an optimization process, the MPC can be affected by local minima, due to the presence of obstacles in the workspace. To overcome this problem, we drive the MPC by computing intermediate waypoints on the basis of the current position of the tools, the position of the obstacles and the target position provided by the *Intelligence module*. Compared to other solutions found in [20], [21], this approach has the advantage of being specifically optimized for the guidance of laparoscopic tools in a shared control environment with dynamical constraints. Operations can, therefore, be performed in a tight environment by continuously searching for optimized solutions given both collision-free geometrical constraints over moving obstacles and varying velocities to respond to uncertainties.

III. DESIGN OF THE MODEL PREDICTIVE CONTROLLER

Safety-critical scenarios, of which surgical theaters are a prime example, require to enforce constraints on velocity limits and collision avoidance, that are taken into account to design our MPC-based control system. At the same time, the control system has to deal with the uncertainty in the recognition of the action that derives by the Intelligence module. This uncertainty is the residual of the confidence level $\alpha \in [0, 1]$ paired to each action identified by the *Action Recognition* system within the *Intelligence module* [16]. We use the confidence level to modulate the Cartesian velocity of the robots while the control system is planning the motion of the robot towards the goal position. The intent is to avoid abrupt movements towards possibly incorrect goal positions, which are, on average, associated with low confidence levels, while avoiding stalling the system and impeding the motion of the teleoperated tools. As the confidence increases for correct predictions, the modulated tool velocity will consequently increase as well.

A. Robot-Obstacle distance computation

Since a safe interaction between the robotic systems and the patient needs always to be guaranteed, collisions between the two *controlled tools* and between the *controlled tools* and the *obstacles* have necessarily to be avoided.

We decide to model each tools in the workspace using virtual capsules, in order to enclose them into the fittest and simplest shape. This allows to make the computation of the relative distance among the objects very simple and fast [22]. Given a pair of Cartesian points, a capsule is an object composed by two hemispheres, centered in that points, and a cylinder, with longitudinal axis linking the two points. Let i and j be two generic tools into the workspace. The distance between the two capsules that enclose the i-th and j-th tool can be defined as

$$d_i^j = d_{ax_i}^{ax_j} - r_i - r_j \tag{1}$$

where $d_{ax_i}^{ax_j}$ is the distance between the axes of the capsules, computed as the distance between segments shown in [22], while r_i and r_j are the radii of the i-th and j-th tool virtual capsule. Fig. 2 shows the procedure used to build a capsule around a laparoscopic tool. Exploiting (1) the distances between the tools can be easily computed and embedded into the MPC to avoid collisions.

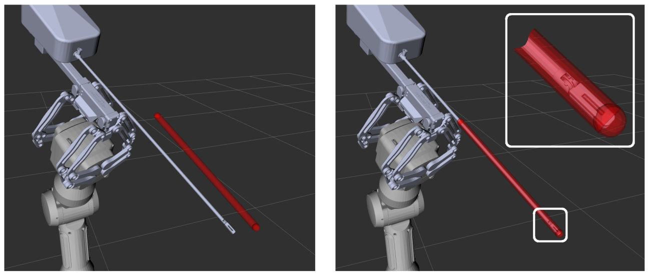


Fig. 2. Procedure used to enclose a tool into a capsule. On the left, the CAD model of the robotic laparoscopic tool and, in red, a capsule. On the right, the capsule wrapping the robotic laparoscopic tool.

B. Robot model

In the following we will consider as *controlled tools* two 4-DOFs robotic laparoscopic tools with a RCM. Since the movements are performed with the tools tips closed for safety reasons, the 4th degree of freedom of the robot, i.e. the rotation along its axis, does not affect the obstacle avoidance, so we do not consider it in the MPC formulation. We indicate with $\bar{x}_j \in \mathbb{R}^3, j \in \{r, l\}$ the coordinates of the position of the end-effector as the state of each controlled tool, with the subscripts r and l to identify the right and the left tool. Let $\bar{u}_j \in \mathbb{R}^3$ be the control input, where $\bar{u} \in \mathbb{R}^3$ are the input linear velocities of the end-effector. The overall system can be kinematically modeled as a single integrator in the discrete time domain:

$$x(k+1) = x(k) + Bu(k) \tag{2}$$

where $x = [\bar{x}_r, \bar{x}_l] \in \mathbb{R}^6$ is the state vector comprising the two tools, $B = \text{diag}\{\Delta t_c\} \in \mathbb{R}^{6\times 6}$ is the input matrix, Δt_c is the sampling time ($t = k\Delta t_c$, $k \in \mathbb{Z}$) and $u = [\bar{u}_r, \bar{u}_l] \in \mathbb{R}^6$ represents the control input.

C. Constraints

Two types of constraints are considered in the MPC formulation:

- *Velocity limit*: The velocities of the robots are physically limited. This can be translated into a bound on the control input:

$$||\bar{u}_j(k)|| \leq \alpha(k)\, \bar{u}_j^{\max} \tag{3}$$

where $\bar{u}_j^{\max} \in \mathbb{R}^+$ is the linear velocity limit and $\alpha(k)$ is the confidence level used to modulate the velocity of the robots depending on the uncertainty in the action recognition.
- *Collision avoidance constraint*: As introduced in Section III-A, collisions between the tools into the workspace need to be avoided during the task execution. To this aim, the distances between capsules computed as (1) are exploited and the collisions between the i-th tool and the j-th tool at time instant k are avoided by setting the following constraint:

$$d_i^j(k) \geq d_s \tag{4}$$

where $d_s \in \mathbb{R}$ is a user-defined positive parameter representing the safety distance.

D. Cost function

The cost function of the MPC is defined as:

$$J(x^{\text{MPC}}, x) = \sum_{i=0}^{p-1} ||x^{\text{MPC}}(k) - \hat{x}(k+i)|| \tag{5}$$

where $p = N/\Delta t_c$ is the number of time steps in the prediction horizon N for the MPC sampling time Δt_c, $x^{\text{MPC}}(k) \in \mathbb{R}^6$ is the desired state at the time step k driven to the MPC, and $\hat{x}(k+i) \in \mathbb{R}^6$ is the predicted state with initial condition $\hat{x}(k+0) = x(k)$. This cost function allows the system to reach the deisred position with a straight trajectory if no obstacles are detected. The desired state of the MPC $x^{\text{MPC}}(k)$ will be provided by the planner illustrated in Section IV.

The rotation and the velocity of the tool along its axis are controlled by an external proportional controller. This separation between linear and rotational control is possible by the nature of the 4-DOFs standard laparoscopy tools and the policy of moving the robots keeping the instruments closed.

E. MPC formulation

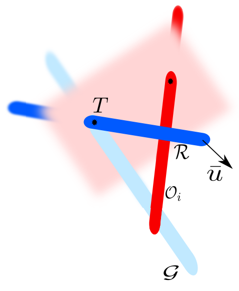
The solution of the following constrained finite-horizon optimal control problem

$$\begin{array}{ll} \min\limits_{\mathbf{u}} & \sum_{i=0}^{p-1} ||x^{\text{MPC}}(k) - \hat{x}(k+i)|| \\ s.t. & \hat{x}(k+i+1) = \hat{x}(k+i) + Bu(k+i) \\ & ||u_j(k)|| \leq \alpha(k)\, u_j^{\max} \\ & d_{r_r}^{r_l}(k+i) \geq d_s \\ & d_{r_j}^{o_h}(k+i) \geq d_s \\ & i = 0, ..., p-1 \\ & h = 0, ..., N_o \\ & j \in \{r, l\} \end{array} \tag{6}$$

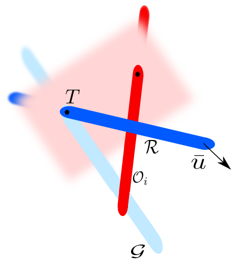
returns the optimal control input sequence $\mathbf{u} = [u(k), ..., u(k+p-1)]$. In the optimization problem, $\hat{x}(k+i+1)$ represents the estimation of the state at time $k+i+1$ computed using the model (2), $u(k+i)$ is the linear control input at time $k+i$ and $d_{r_r}^{r_l}(k+i)$ is the distance between the virtual capsule built around the *controlled tools* at time $k+1$. $d_{r_j}^{o_h}(k+i)$ is the distance between the virtual capsules built around the *controlled tool* j and the *obstacle* h at time $k+1$ with N_o the number of obstacles in the workspace. The position of each one of the N_o *obstacles* in the prediction horizon is computed considering its velocity to be constant over the entire horizon. The same holds for the desired position $x^{\text{MPC}}(k)$ and the confidence level $\alpha(k)$. Finally, the first component $u(k)$ is used to compute the desired motion position $x^d(k+1) \triangleq x(k) + Bu(k)$ to be actually reached by the robots.

IV. WAYPOINTS GENERATION

When the target configuration, that we can denote as x^g, selected by the *Intelligence module* shown in Section II is directly provided as the MPC desired position $\bar{x}^{mpc}$, the MPC alone may not guarantee that the desired configuration is reached. As shown in Fig. 3, it is possible for the solver

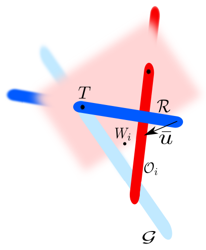


(a)

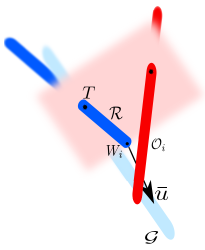


(b)

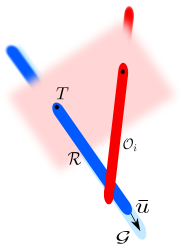
Fig. 3. Tool position (blue), desired tool position (light-blue) and capsule-shaped obstacle (red) before (a) and after (b) the insertion movement. The obstacle cannot be overtaken.



(a)



(b)



(c)

Fig. 4. Tool position (blue), desired tool position (light-blue) and capsule-shaped obstacle (red). The waypoint is used to make one of the arm avoiding the obstacle along its path.

to incur in a local minimum solution: all the constraints are satisfied, but the final target is not reached. This is mainly due to the constraint introduced by the trocar point T the tool is forced to pass through. A possible solution to this problem consists in properly planning a set of waypoints towards the final target configuration to be provided as intermediate goals to the MPC (see Fig. 4). In this way, as shown in [23], it is possible to lead the robot along preferential directions to avoid the obstacle while taking into account the trocar constraint. The proposed planning strategy is based on simple geometric considerations and it can be executed in real time with reduced computational overhead when compared to potential energy-based methods like [20], which is crucial for having a reactive behavior of the robotic system holding the laparoscopic tool. The motion of each arm is planned separately. When the motion of one arm is computed, the other arms are considered as obstacles. The arm holding the tool as well as all the obstacles are wrapped into capsules and, therefore, the planning problem can be seen as the problem of planning the motion of one capsule constrained to a trocar point from an initial to a final configuration while avoiding other capsules. In the following, we will denote with $\mathcal{C}(C_1, C_2)$ a generic capsule, where $C_1 = (x_{C_1}(t), y_{C_1}(t), z_{C_1}(t))$ and $C_2 = (x_{C_2}(t), y_{C_2}(t), z_{C_2}(t))$ are the two end-points of the capsule that identify the pose of the capsule; $\overline{C_1C_2}$ denotes the axis passing through C_1 and C_2. We will omit the end-points, when clear from the context.

Let $\mathcal{R}(R_1, R_2)$, $\mathcal{G}(G_1, G_2)$ and $\mathcal{O}_i(O_{1,i}, O_{2,i})$, with $i = 0, \ldots, N_o$ be the capsules identifying the tool whose motion needs to be planned, the goal configuration where $\mathcal{R}$ has to be taken, and the obstacles, respectively. The planning algorithm is reported in Alg. 1. The necessary data are the capsules of the arm $\mathcal{R}$, of the goal $\mathcal{G}$ and of all the obstacles $\mathcal{O}_i$, $i = 1, \ldots, N_o$. For each obstacle, a local waypoint is generated according to the following procedure. The motion plane $\mathscr{M}$, i.e. the plane on which the tool can reach the goal configuration in case of no obstacles, is generated (Line 3). Formally, this is the plane orthogonal to the normal vector

$$n = \frac{R_1 - R_2}{\|R_1 - R_2\|} \times \frac{G_1 - G_2}{\|G_1 - G_2\|} \tag{7}$$

and containing $\overline{R_1R_2}$ and $\overline{G_1G_2}$. Then a set S_i of possible escape points from the obstacle is generated by uniformly sampling the space around the endpoint of the obstacle capsule $\mathcal{O}_i$ that is the closest to the tool (Line 5). Among these points, Z_i, the closest to the trocar, is chosen (Line 6) in order to give a preference to the retraction of the tool, which is often a feasible option. If the capsule of the obstacle is parallel to the motion plane, i.e. $(O_{i,1} - O_{i,2}) \cdot n = 0$, but not contained in it, or if the capsule is not intersecting the plane, i.e. $((O_{i,1} + \sigma(O_{i,2} - O_{i,1})) - R_1) \cdot n = 0$ for all $\sigma \in [0, 1]$, then the motion plane is free and the robot can reach the desired configuration. Thus, no local waypoints are generated (Line 8). If the obstacle intersects the motion plane, a local waypoint W_i is generated by projecting Z_i on the motion plane along the obstacle axis (Line 9). In this way, W_i is reachable by the robot and it does not intersect the obstacle by construction. The set containing the local waypoints is updated (Line 10).

The global waypoint is computed as the centroid of the waypoints associated to each obstacle, using as weight of each waypoint β_i the inverse of the distance d_i between the tool and the related obstacle (Line 11):

$$W = \frac{\sum_{i=1}^{N} \beta_i W_i}{\sum_{i=1}^{N} \beta_i} \tag{8}$$

where N represents the cardinality of $\mathscr{Q}$ and $\beta_i = 1/d_i$. Despite the fact that each local waypoint is external to the related obstacle, it is possible (though unlikely) that their centroid, computed as (8), could lie inside one of the obstacles: in this case, the waypoint W is set equal to the

Algorithm 1: Local waypoint strategy

```
1  Data: $\mathcal{R}$, $\mathcal{G}$,$\mathcal{O}_i$
2  $\mathscr{Q} = \emptyset$
3  $\mathscr{M}$=computeMotionPlane($\overline{R_1R_2}$,$\overline{G_1G_2}$)
4  for $i \leftarrow 1$ to $N_o$ do
5      $S_i$ = Sample($O_{1,i}, O_{2,i}$)
6      $Z_i$=getClosest($S_i$, $T$)
7      if (isFree($O_{i,1}, O_{i,2}, R_1, \mathscr{M}$)) then
8          $W_i$=$\emptyset$)
       else
9          $W_i$=project($Z_i, \mathscr{M}, \overline{O_{i,1}O_{i,2}}$)
10     addWayPoint($W_i$, $\mathscr{Q}$)
11 $W$=computeFinalWayPoint($\mathscr{Q}$)
```

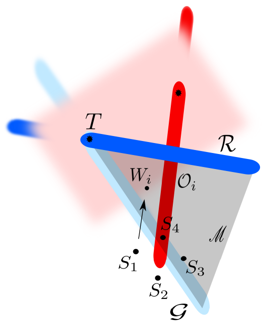


Fig. 5. Tool position (blue), desired tool position (light-blue) and capsule-shaped obstacle (red). The point S_1 is chosen to be projected on the plane $\mathscr{M}$ (grey) to find the waypoint W_i

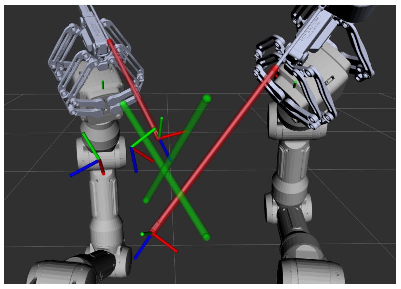


Fig. 6. Simulation environment. A SARAS arm model (courtesy of Medineering GmbH) is placed on the right side and on the left side. In red, the virtual capsules wrapping each arm's tool. In green, the virtual capsules wrapping the obstacles. The frames at the end of each arm represent the pose of the end-effector while the other two frames represents the goal positions.

trocar, thus the MPC will compute an extraction movement, which is always collision free.

V. SIMULATIONS AND EXPERIMENTS

The validation of the proposed system has been performed on the SARAS platform. In particular, four laparoscopic tools operate in a shared workspace: two of them are mounted on prototype robots (Medineering GmbH) and controlled by the autonomous system, while the other two (the da Vinci® arms) are teleoperated and represent the obstacles to be avoided by the autonomous ones. The control system proposed in this paper moves the controlled tools in order to avoid collisions between all the tools inside the environment.

A. Validation in simulation

At first, a simulation environment has been developed to perform preliminary tests. In order to simulate the real movement of the robot and faithfully reproduce the real setup, a visual model and a kinematic simulator of the SARAS arms were created using the 3D visualizer ROS Rviz and the CAD's provided by the robot's company. The arms were simulated introducing in the simulation environment the position of the RCM and the position of the end-effector of two arms, with the relative virtual capsules. Figure 6 shows the simulation environment. Simulations are performed providing to the system the goal configuration x_g, the initial configuration of the two arms $\tilde{x}_r, \tilde{x}_l$ and the initial configuration of the two obstacles. We set the SARAS and da Vinci® capsules' radii to $7.5\,\mathrm{mm}$, and the additional safety distance between capsules $d_s = 5\,\mathrm{mm}$; we conservatively set the maximum tool velocities to $u_j^{\max} = 10\,\frac{\mathrm{mm}}{\mathrm{s}}$. Figure 7 shows the results achieved using the simulation environment. The norm of the Cartesian velocity (Fig. 7-a) of the two controlled arms clearly shows the modulation introduced by the confidence level, reported in Fig. 7-b, underlining the capability of the controller of scaling the velocities if an uncertain situation is detected (e.g. at time $t = 13s$). Figure 7-c shows the evolution of the Cartesian positions. For the sake of clarity, only movements along one of the axis are reported (here and in all the subsequent plots). The position of the waypoint switches during the simulation in order to allow the SARAS arms to overtake the obstacles. This happens only if an obstacle needs to be overtaken, as for the left robot in the time interval $t \in [0, 25]$. If no obstacle needs to be overtaken, the waypoint is set to the target position, as for the right robot, where the waypoint and the target position overlap for the entire simulation. Since the waypoint position is used as reference for the MPC controller and the waypoint position converge to the target position, the overall controller allows the system to reach the target position. The real Cartesian position of the arms is not reported in the plots since the implemented simulator is purely kinematic, namely, there are no differences between the commanded position and the real position. Thanks to the MPC controller and the waypoint motion strategy, the controller is able to perform all the movements avoiding collision between tools, as clearly visible in Fig. 7-d where the distances between the tools are reported. A particular behavior of the controller can be also observed from Fig. 7. Indeed, in the first 30 seconds of the simulation, the right robot reaches the target position and starts to track it, as visible in Fig. 7-c. At that time, the right robot moves in order to allow the left robot to reach its goal position. Indeed, the distance between the two arms goes to the minimum allowed distance, as visible in Fig. 7-d. Finally, a new configuration is computed for both the robots in order to minimize the distance from the target position. Simulation tests have been performed with moving obstacles (Figure 8) to validate the local waypoint algorithm and the response of the MPC optimization. The obstacles pivot at a constant tool-tip linear velocity of $2\,\frac{\mathrm{mm}}{\mathrm{s}}$ to interfere with the initial planned waypoint; the different geometrical alignment of the tools forces the re-evaluation of a new waypoint. A direct comparison of Figure 7 and 8, subfigures (c) and (d), illustrates the adapted control strategy to the moving obstacles as the obstacle closes in to the moving tools, thus forcing a different trajectory. Both the static and moving obstacles behaviors are present in the videoclip attached to this paper.

B. Validation on the SARAS setup

The experiments has been performed providing to the system the goal configuration x_g. The configuration of the

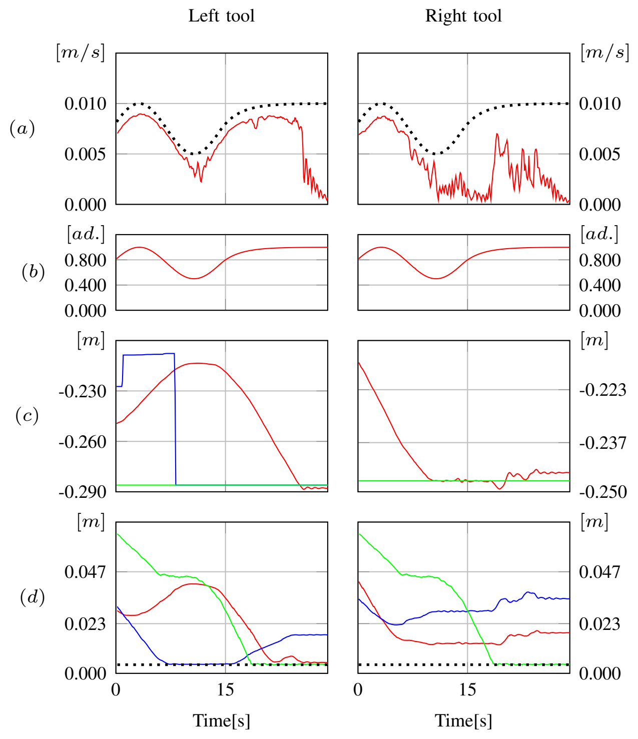


Fig. 7. Simulation results. (a) The norm of the controller output velocity (red line) and the confidence modulated maximum velocity norm (black dotted line). (b) The confidence level. (c) The commanded Cartesian position of the robot (red line), the Cartesian position of the waypoint (blue line) and the Cartesian position of the target position (green line). (d) The distance between the robot tool capsule and the first obstacle capsule (red line), the distance between the robot tool capsule and the second obstacle capsule (blue line), the distance between the robots tool capsules (green line) and the minimum allowed distance between tools (black dotted line). Plots are reported for both left and right robots.

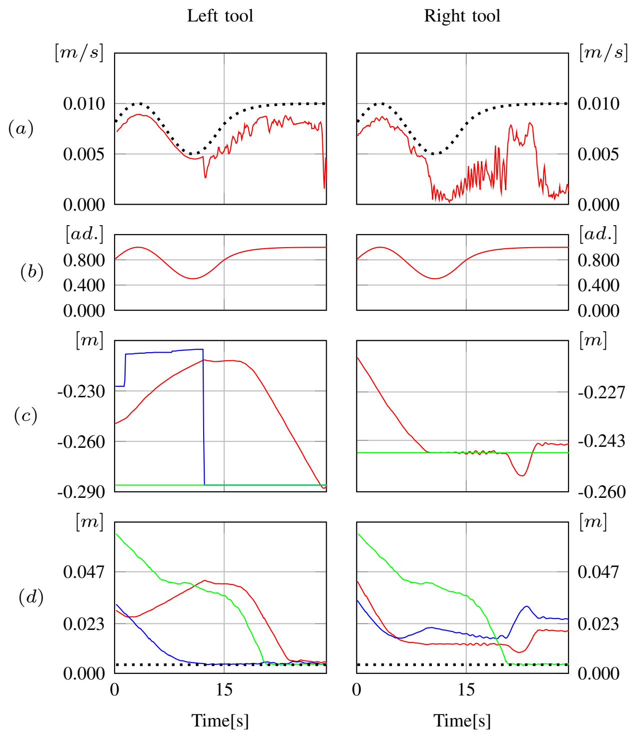


Fig. 8. Simulation results with moving obstacles (refer to the caption for Figure 7).

two arms $\tilde{x}_r, \tilde{x}_l$ and the configuration of the two obstacles are continuously updating using the robots readings. The two controlled arms and the two obstacles arms are initialized in such a way that each controlled arm needs to overcome an obstacle, in order to highlight the capabilities of the controller to avoid obstacles. The radius of the capsules are set to 5 mm for the da Vinci® tools and to 4 mm for the SARAS tools to fit their dimensions with a little safety margin. The safety distance d_s is set to 1 cm, higher than in simulated environment to take into account possible calibration inaccuracies. Figure 9 shows the results achieved using the real setup and confirms the results obtained in simulation. Figure 9-a reports the Cartesian velocities of the two controlled arms while Fig. 9-c reports their Cartesian positions. It is worth highlighting that the noise in the velocities in Fig. 9-a is due both to the numerical derivation of positions measured by potentiometers (and not encoders) and by the fact that the RCMs are virtually (via software) but not physically present. With real trocars the shaking of such slender (and not collocated) tools would be drastically reduced. Good tracking performances can be appreciated looking at the small difference between the commanded Cartesian position and the real Cartesian position (red lines and orange lines in Fig. 9-c). This shows that the robots implementing the MPC commands reach their target positions while avoiding the obstacles, thanks to the waypoint motion strategy described in Section IV. All the movements are performed avoiding collisions, as clearly appreciable in Fig. 9-d, and modulating the velocities with respect to the confidence level provided to the Control module. The same position was commanded as goal configuration for the two controlled tools. Referring to Fig. 9, the position of the waypoints is computed as an intermediate point for both the controlled arms. The controlled tools move to the waypoints. When the tool reaches the waypoint than the waypoint switches to the target position, driving the robot towards the goal configuration since obstacles are assumed to have been already overcome. Since the goal configuration is the same for both the tools, neither of them reaches the target position since a collision would occur. The system converges to a configuration where the distance between the actual position and the desired one is minimized but collisions are avoided, as observed also in simulation.

VI. CONCLUSIONS

In semi-autonomous R-MIS a safe interaction between the robotic system and the patient needs to be guaranteed. Thus, collisions between all the tools inside the workspace and between the tools and the obstacles need to be avoided.

In this paper, we proposed a reactive approach to motion planning to be applied in R-MIS, where multi-arms laparoscopic surgical robots may be present into a shared dynamic environment. The motion planner is based on a

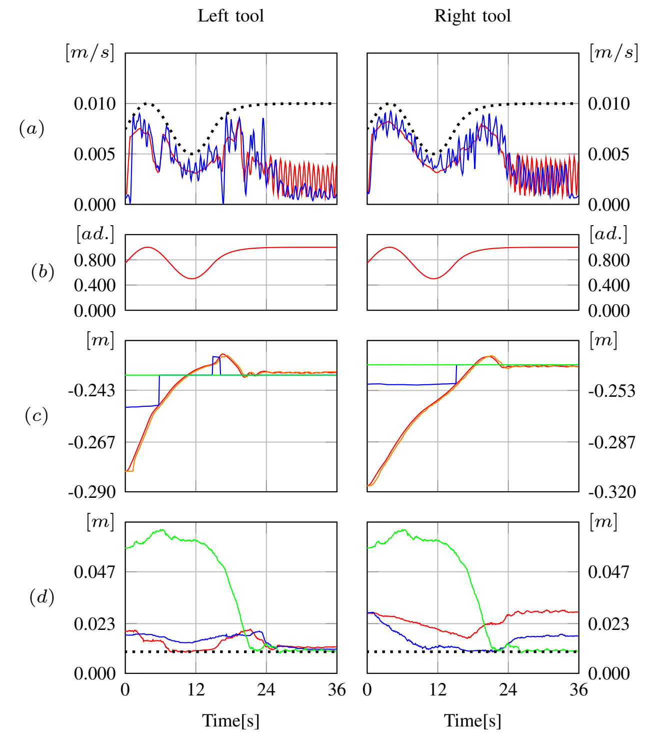


Fig. 9. Experimental results. (a) The norm of the controller output velocity (red line), the norm of the real velocity of the robot (blue line) and the confidence modulated maximum velocity norm (black dotted line). (b) The confidence level. (c) The commanded Cartesian position of the robot (red line), the Cartesian position of the waypoint (blue line), the real Cartesian position of the robot (orange line) and the Cartesian position of the target position (green line). (d) The distance between the robot tool capsule and the first obstacle capsule (red line), the distance between the robot tool capsule and the second obstacle capsule (blue line), the distance between the robots tool capsules (green line) and the minimum allowed distance between tools (black dotted line). Plots are reported for both left and right robots

Model Predictive Controller that allows to rapidly converge towards the optimal solution, while satisfying all the constraints for velocity limits and collision avoidance. Moreover, an innovative strategy has been proposed for computing waypoints that allows to overtake the obstacles when the MPC alone would fail. The proposed control strategy has been validated on a realistic surgical scenario.

VII. Acknowledgment

The authors would like to thank Johann Wigger, Sabine Hertle and Medineering GmbH for the development of the SARAS robots used in the experiments.

References

[1] R. H. Taylor, A. Menciassi, G. Fichtinger, P. Fiorini, and P. Dario, “Medical robotics and computer-integrated surgery,” in *Springer handbook of robotics*. Springer, 2016, pp. 1657–1684.

[2] G.-Z. Yang, J. Cambias, K. Cleary, E. Daimler, J. Drake, P. E. Dupont, N. Hata, P. Kazanzides, S. Martel, R. V. Patel, V. J. Santos, and R. H. Taylor, “Medical robotics – Regulatory, ethical, and legal considerations for increasing levels of autonomy,” *Science Robotics*, vol. 2, no. 4, p. 8638, 2017.

[3] T. Haidegger, “Autonomy for surgical robots: Concepts and paradigms,” *IEEE Transactions on Medical Robotics and Bionics*, vol. 1, no. 2, pp. 65–76, May 2019.

[4] F. Ficuciello, G. Tamburrini, A. Arezzo, L. Villani, and B. Siciliano, “Autonomy in surgical robots and its meaningful human control,” *Paladyn Journal of Behavioral Robotics*, vol. 10, no. 1, pp. 30–43, 2019.

[5] N. Preda, F. Ferraguti, G. De Rossi, C. Secchi, R. Muradore, P. Fiorini, and M. Bonf, “A cognitive robot control architecture for autonomous execution of surgical tasks,” *Journal of Medical Robotics Research*, vol. 1, no. 04, 2016.

[6] F. Ferraguti, N. Preda, M. Bonfè, and C. Secchi, “Bilateral teleoperation of a dual arms surgical robot with passive virtual fixtures generation,” in *IROS*, 2015.

[7] N. Preda, A. Manurung, O. Lambercy, R. Gassert, and M. Bonfè, “Motion planning for a multi-arm surgical robot using both sampling-based algorithms and motion primitives,” in *IROS*, 2015.

[8] F. Ferraguti, N. Preda, A. O. Manurung, M. Bonfè, 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.

[9] C. Reiley, E. Plaku, and G. Hager, “Motion generation of robotic surgical tasks: Learning from expert demonstrations,” in *EMBC*, 2010.

[10] J. Schulman, A. Gupta, S. Venkatesan, M. Tayson-Frederick, and P. Abbeel, “A case study of trajectory transfer through non-rigid registration for a simplified suturing scenario,” in *IROS*, 2013.

[11] H. Wang, S. Wang, J. Ding, and H. Luo, “Suturing and tying knots assisted by a surgical robot system in laryngeal MIS,” *Robotica*, vol. 28, no. SI 02, pp. 241–252, 2010.

[12] F. Nageotte, P. Zanne, C. Doignon, and M. deMathelin, “Stitching planning in laparoscopic surgery: Towards robot-assisted suturing,” *International Journal of Robotics Research*, vol. 28, no. 10, pp. 1303–1321, 2009.

[13] I. Elgezua, Y. Kobayashi, and M. G. Fujie, “Survey on current state-of-the-art in needle insertion robots: Open challenges for application in real surgery,” *Procedia CIRP*, vol. 5, pp. 94–99, 2013.

[14] 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 *ISMR*, 2019.

[15] 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 *ISMICT*, 2019.

[16] G. De Rossi, M. Minelli, A. Sozzi, N. Piccinelli, F. Ferraguti, F. Setti, M. Bonfè, C. Secchi, and R. Muradore, “Cognitive robotic architecture for semi-autonomous execution of manipulation tasks in a surgical environment,” in *IROS*, 2019.

[17] 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 *ICRA*, 2019.

[18] S. M. Khansari-Zadeh and A. Billard, “A dynamical system approach to realtime obstacle avoidance,” *Autonomous Robots*, vol. 32, no. 4, pp. 433–454, 2012.

[19] M. Cefalo, E. Magrini, and G. Oriolo, “Sensor-based task-constrained motion planning using model predictive control,” in *SYROCO*, 2018.

[20] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, “Finding locally optimal, collision-free trajectories with sequential convex optimization.” in *Robotics: science and systems*, vol. 9, no. 1. Citeseer, 2013, pp. 1–10.

[21] B. Kehoe, G. Kahn, J. Mahler, J. Kim, A. Lee, A. Lee, K. Nakagawa, S. Patil, W. D. Boyd, P. Abbeel, and K. Goldberg, “Autonomous multilateral debridement with the raven surgical robot,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 1432–1439.

[22] V. J. Lumelsky, “On fast computation of distance between line segments,” *Information Processing Letters*, vol. 21, pp. 55–61, 1985.

[23] A. Sozzi, M. Bonfè, S. Farsoni, G. De Rossi, and R. Muradore, “Dynamic motion planning for autonomous assistive surgical robots,” *Electronics*, vol. 8, p. 957, 2019.