









Learned Task Space Control to Reduce the Effort in Controlling Redundant Surgical Robots

Murali Karnam¹(✉) , Manuela Eugster¹ , Riccardo Parini^{1,3},
Philippe C. Cattin² , Elena De Momi³ , Georg Rauter¹ ,
and Nicolas Gerig¹ 

- ¹ BIROMED-Lab, Department of Biomedical Engineering, University of Basel,
4123 Allschwil, Switzerland
murali.karnam@unibas.ch
- ² CIAN, Department of Biomedical Engineering, University of Basel,
4123 Allschwil, Switzerland
- ³ NearLab Medical Robotics, Department of Electronics, Information
and Bioengineering, Politecnico di Milano, 20133 Milan, Italy
<https://biomed.dbe.unibas.ch>

Abstract. Redundant robots allow multiple robot joint configurations for the same end-effector pose by moving only in null space. Robot's motions in null space are not intuitive to predict in general and in particular for medical personnel. In this work, we present a control concept that allows the operator to focus on the correct end-effector pose during time-critical tasks, e.g. change of the endoscope pose during a surgical intervention, while the shape of the redundant robotic structure is handled autonomously based on previously learnt preferred shapes close to the actual end-effector pose. We investigated the benefit of the proposed *learned task space control* over *naive task space control* that required an operator to manually control a virtual robot in task space and null space independently. In a first user study, we found that *learned task space control* significantly reduced the effort – as measured by task duration and task load – for operators compared to *naive task space control*.

Keywords: Redundancy · Null space · Control · Medical robots

1 Introduction

Redundant robots have more degrees of freedom (DoF) than required for a primary task (end-effector pose control). Null space of redundant robots enables changing the robot shape (joint configuration) as a secondary task with lower priority, and not affect the primary task [1]. Secondary tasks could for example be avoiding obstacles or joint limits. Typical methods to control robots for

secondary tasks may rely on mathematically defined objective functions [2,3], making null space motion challenging to predict for medical personnel. Alternative methods take user input to control the null space [4,5] and are considered more intuitive since the user has complete control over the robot motion. Nevertheless, methods based on user input may increase the overall effort for the operator as both task space and null space must be controlled simultaneously.

We aim to increase the intuitiveness in controlling redundant robots and surgical robots in particular by decreasing the effort for medical personnel in handling non-intuitive motion of the redundant DoF of the robot. We achieve this by recording the desired robot joint configurations during preoperative planning. While the operator controls the robot intraoperatively to perform the surgery (primary task), the robot automatically moves in null space towards the closest recorded joint configuration (Fig. 1).

To reach our goal, we present a *learned task space control* method, which controls both task space and null space of a redundant robot. We hypothesize that the operator's effort to control a robot is reduced with the proposed control method as compared to a *naive task space control* method where the operator actively controls task space and null space.

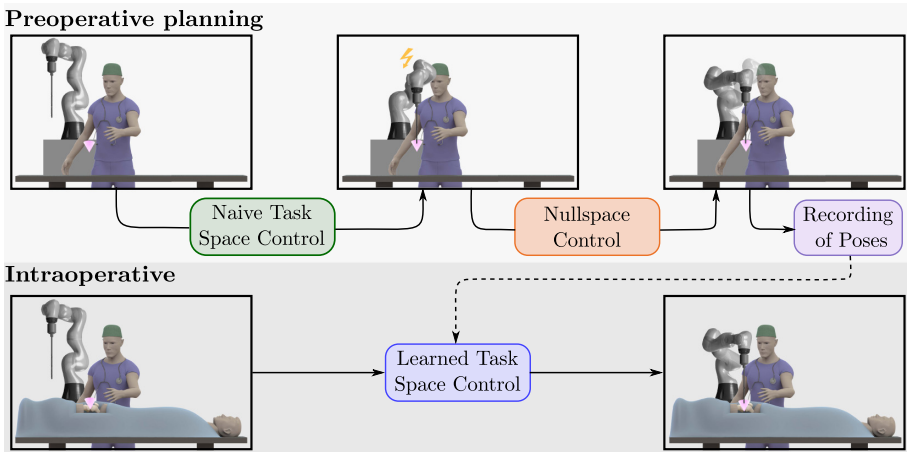


Fig. 1. Workflow overview: during preoperative planning, the operator uses *naive task space control* to move the robot to reach a desired end-effector pose (pink cone). At the desired end-effector pose, the robot's joint configuration can be reshaped in null space. The operator can record the current end-effector pose and related joint configuration. Intraoperatively, the operator can move the robotic arm using *learned task space control* to any desired end-effector pose. As the operator controls the robot in task space, the robot's joints automatically move towards the closest recorded joint configuration.

2 Materials and Methods

2.1 Apparatus

A virtual 7-DoF redundant robot (KUKA LBR iiwa, KUKA AG, Augsburg, Germany) with a rigid endoscope as the end-effector was visualized on a screen. This end-effector was controlled with a table-fixed force/torque (F/T) sensor with a handle on top (Fig. 2).

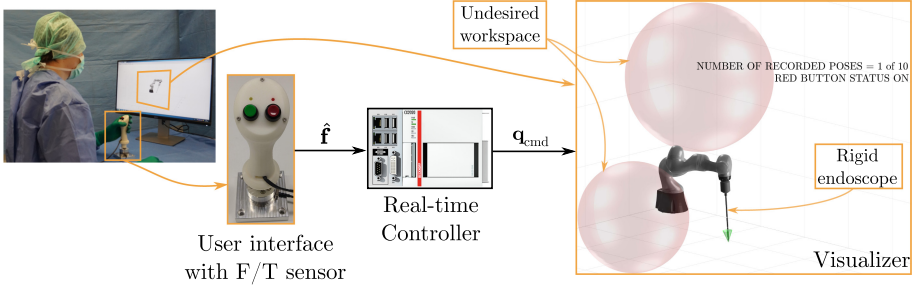


Fig. 2. Schematic of the setup. A handle with two buttons allows switching control modes and enables moving to robot poses by applying force on a F/T sensor (Mini45, ATI Industrial Automation). Real-time calculations are performed on a CPU (CX2020, Beckhoff Automation GmbH & Co. KG, Verl, Germany) that reads the forces ($\hat{\mathbf{f}}$) that the user applies to the handle, and calculates the corresponding desired joint configuration \mathbf{q}_{cmd} . A GUI shows the robot configuration to the user. In the GUI, obstacle regions are indicated as red spheres and the target pose for the robot’s end-effector is shown as a cone. Once the end-effector has reached the target pose, the color of the cone is changed from pink to green. The status of the handle buttons, the number of existing recorded poses, and the robot status when it reaches a work-, joint-, or null space limit are shown with text to the user.

2.2 Control Modes

The two buttons on the handle (Fig. 2) allow the user recording poses and switching between different control modes – null space control, *naive task space control*, and *learned task space control* (Fig. 3). Measured user forces ($\hat{\mathbf{f}}$) were filtered using a moving median filter with a 100 ms time window.

Record Pose. The interface allows a user to record the current estimated end-effector pose and joint configuration as $(\mathbf{x}_{\text{rec}_i}, \mathbf{q}_{\text{rec}_i})$. The pairs of all recorded poses (\mathbf{X}_{rec}) and configurations (\mathbf{Q}_{rec}), are used in *learned task space control*. If an end-effector pose is recorded that is too close to an already recorded one, the previous pose is overwritten with the new pose and joint configuration. Two poses (\mathbf{x}_a and \mathbf{x}_b) were considered too close if the dissimilarity metrics in position

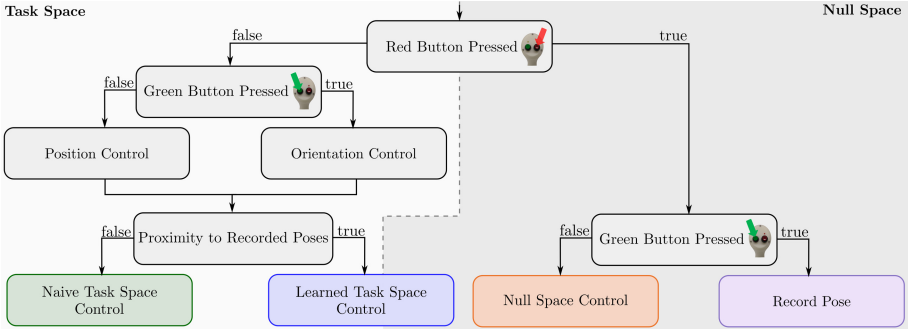


Fig. 3. User control modes overview: the red button allows the user to switch between task space and null space control. The green button allows the user to record the robot pose and joint configuration used for *learned task space control*.

(Euclidean distance [6]) and orientation (deviation of the rotation matrix from the identity matrix [7]) between them is below a predefined threshold:

$$\begin{aligned} \|\Delta \mathbf{x}_{\text{pos}}\|_2 &< 0.15 \text{ m}, \text{ and} \\ \|\mathbf{I} - {}^a \mathbf{R}_b\|_F &< 0.73, \end{aligned} \quad (1)$$

where \mathbf{x}_{pos} and ${}^a \mathbf{R}_b$ are the position vector and the rotational transformation matrix between the two end-effector poses (\mathbf{x}_a and \mathbf{x}_b), \mathbf{I} is a 3×3 identity matrix, and $\|\cdot\|_F$ denotes the Frobenius norm of the matrix.

Null Space Control. Based on the force applied by the user, \mathbf{q}_{cmd} is calculated such that the end-effector pose is unchanged, while the joint configuration changes only in the null space. The details have been presented previously [8].

Naive Task Space Control. The commanded robot joint configuration \mathbf{q}_{cmd} is calculated based on the estimated joint configuration $\hat{\mathbf{q}}$, and the desired end-effector velocities in task space $\dot{\mathbf{x}}_{\text{cmd}}$ (Fig. 4, left):

$$\mathbf{q}_{\text{cmd}} = \hat{\mathbf{q}} + T_s \cdot \mathbf{J}^+ \cdot \dot{\mathbf{x}}_{\text{cmd}}, \quad \mathbf{J}^+ = \mathbf{J}^T \left(\mathbf{J} \mathbf{J}^T - \lambda^2 \mathbf{I} \right)^{-1} \quad (2)$$

where \mathbf{J}^+ is the damped Moore-Penrose pseudo-inverse of the Jacobian, with the damping parameter $\lambda = 0.001$, and $T_s = 0.001 \text{ s}$ is the cycle time of the real-time system. The desired end-effector pose velocity, $\dot{\mathbf{x}}_{\text{cmd}}$, consists of velocities in position (\mathbf{v}_{cmd}) and orientation ($\boldsymbol{\omega}_{\text{cmd}}$). They are calculated from the measured user forces as, $\mathbf{v}_{\text{cmd}} = c_v \cdot \hat{\mathbf{f}}$ for position control and $\boldsymbol{\omega}_{\text{cmd}} = c_\omega \cdot \hat{\mathbf{f}}$ for orientation control, where $c_v = 1.4 \frac{\text{m}}{\text{Ns}}$ and $c_\omega = 1.0 \frac{\text{rad}}{\text{Ns}}$ were used. The commanded velocity was saturated to 0.035 m/s in position and 0.026 rad/s in orientation.

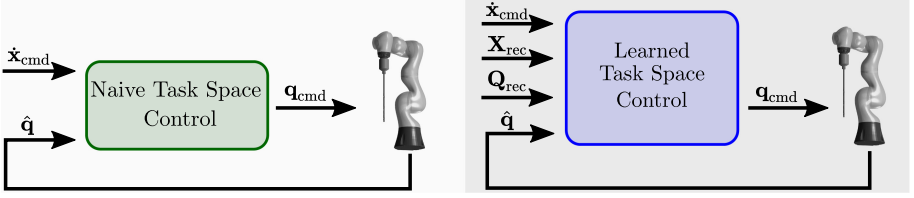


Fig. 4. Task space control modes: *naive task space control* (left) and *learned task space control* (right). At each time step, the control modes provide a commanded joint configuration \mathbf{q}_{cmd} based on the estimated joint configuration $\hat{\mathbf{q}}$ and commanded end-effector velocities $\dot{\mathbf{x}}_{\text{cmd}}$. The *learned task space control* further considers recorded end-effector poses \mathbf{X}_{rec} and the corresponding robot configurations \mathbf{Q}_{rec} .

Learned Task Space Control. For each time step T_s the *learned task space control* (Fig. 4, right) requires as inputs the commanded end-effector velocity $\dot{\mathbf{x}}_{\text{cmd}}$, the recorded end-effector poses \mathbf{X}_{rec} , the related robot’s joint configuration \mathbf{Q}_{rec} , and the estimated robot’s joint configuration $\hat{\mathbf{q}}$. The output is the resulting commanded joint configuration \mathbf{q}_{cmd} :

$$\mathbf{q}_{\text{cmd}} = \hat{\mathbf{q}} + T_s \cdot (\mathbf{J}^+ \cdot \dot{\mathbf{x}}_{\text{cmd}} + \dot{\mathbf{q}}_{\text{LRN}}), \quad (3)$$

which is a summation of the *naive task space control* joint command and a learned joint velocity ($\dot{\mathbf{q}}_{\text{LRN}}$). The learned joint velocity is calculated as the gradient of an objective function ($f(\mathbf{x}, \mathbf{q})$) projected into null space:

$$\dot{\mathbf{q}}_{\text{LRN}} = \mathbf{N} \cdot \nabla_{\mathbf{q}} f(\mathbf{x}, \mathbf{q}) \Big|_{\substack{\mathbf{x}=\hat{\mathbf{x}} \\ \mathbf{q}=\hat{\mathbf{q}}}}, \quad (4)$$

where $\hat{\mathbf{x}}$ is the estimated end-effector pose and \mathbf{N} is the null space projection matrix. The objective function is formulated such that it has a minimum at the closest recorded pose and joint configuration:

$$f(\mathbf{x}, \mathbf{q}) = \dot{q}_{\text{max}} \cdot \frac{w_1}{\hat{\Phi}_{\text{pos}}(\mathbf{x}_{\text{rec}}^*, \mathbf{x}) + w_2} \cdot ((\mathbf{q}_{\text{rec}}^* - \mathbf{q}) \oslash \mathbf{q}_{\text{range}})^2, \quad (5)$$

where, $\mathbf{x}_{\text{rec}}^*$ is the closest among all recorded end-effector poses as per the dissimilarity metric Eq. (1) and $\mathbf{q}_{\text{rec}}^*$ is its corresponding joint configuration. The weight, $w_1 = 0.07$ m, is set manually to tune the effect of *learned task space control*, and a damping term, $w_2 = 0.06$ m, is used to ensure the function is bounded. A maximum robot joint velocity $\dot{q}_{\text{max}} = 30^\circ/\text{s}$ is used to limit the learned joint velocity and not exceed the maximum allowed joint speeds. The difference between the estimated and closest recorded joint configuration is normalized by the joint ranges of the robot $\mathbf{q}_{\text{range}}$, where the Hadamard division symbol \oslash , denotes element-wise division of the two vectors.

2.3 Participants

Six healthy volunteers (all male, age 25 to 45 years, mean age 29.5 years) participated in the pilot study. The participants did not have prior experience with

the user-interface to control a virtual robot, and they were all researchers at the Department of Biomedical Engineering, University of Basel.

2.4 Experimental Protocol

The objective for the participant was to move the robot from a predefined initial configuration to a goal pose (visualized as a cone) and configuration (no collision with obstacles). The study consisted of three tasks:

Task P (planning). *Naive task space control* followed by null space control was used to complete the objective, and the final robot configuration was recorded.

Task N (naive). The objective was completed using *naive task space control*. The controller was manually set to ignore the recorded poses.

Task L (learned). The objective was completed using *learned task space control*, and the controller used the recorded poses from planning.

The trial began by allowing each participant to use the interface and move the robot freely for five minutes to get acquainted with the system. The participants were instructed to complete the tasks as fast as possible. At the end of each task, the participants were asked to fill a NASA Task Load Index (NASA-TLX) questionnaire. With two task space control modes (N and L), the study had a crossover design with one independent variable, leading to two conditions per participant (N-L or L-N). The participants were randomly divided into two groups. Group 1 performed the tasks in the order P-N-L, and Group 2 in the order P-L-N. Each participant performed the three tasks once.

2.5 Evaluation Metrics

The effort for the user for each task was measured with two metrics; (1) The task duration (s), i.e., the time taken by the participant to complete the task; (2) The NASA-TLX scores between 0 and 100. A one-sided Wilcoxon signed-rank test with a confidence level of 5% was performed independently on task duration and task load. The null hypothesis was that the effort to perform the tasks with *naive task space control* was greater than or equal to the effort for *learned task space control*.

3 Results

Task duration ($p < .03$, $W = 20$, $r = .95$) and task load ($p < .02$, $W = 21$, $r = 1.0$) were significantly reduced ($W_{\text{critical}} = 17$) using *learned task space control* compared to *naive task space control* (Fig. 5).

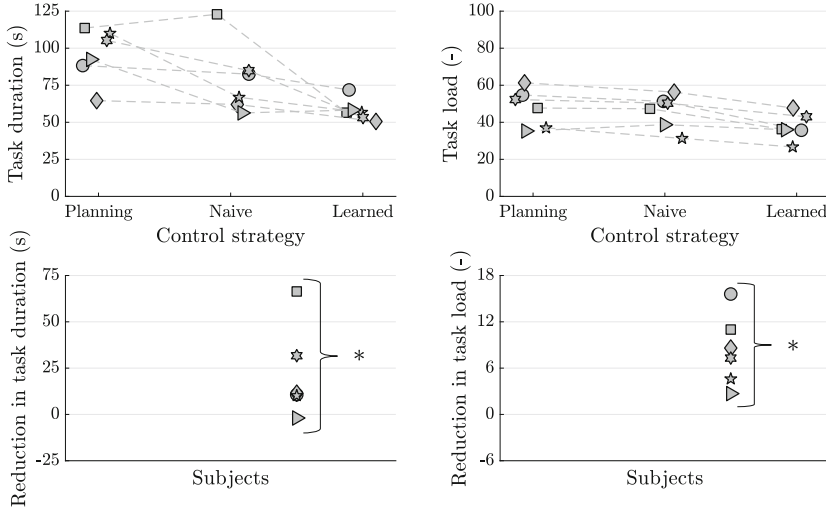


Fig. 5. Task duration (top-left) and task load (top-right); participants are represented by different markers that are connected by a dashed line across the three tasks. The two lower plots show the difference in the task duration (bottom-left) and task load (bottom-right) for each participant between using *naive* and *learned task space control*. The * indicates a significant reduction in the one-sided Wilcoxon signed-rank test.

4 Discussion

The results showed a statistically significant reduction in the effort of participants using *learned task space control* compared to *naive task space control* as evaluated quantitatively by a one sided Wilcoxon signed-rank test. Although the study included only six participants, the pilot study shows that using *learned task space control* for robot control holds potential to reduce medical personnel’s effort during surgery and its utility needs to be investigated with a physical robot in a surgical setting and medical personnel as participants.

We observed that the reduction in the task duration from using *naive task space control* to *learned task space control* was smaller than the time taken to plan and record poses. However, we expect that the robot is re-positioned multiple times in a real intervention, thereby not scaling the time taken to plan the surgery proportionally. Furthermore, the time duration was only used as a measure of effort. Reducing the overall surgery duration was not a goal of this study.

A limitation of the study was the use of a virtual robot visualized on a screen, which might have affected the performance of the users to control the robot in task space. Whether the effort reduces with a physical robot needs to be investigated. We believe that an interface directly mounted on the robot would reduce the effort to control the robot in all modes and make all modes more intuitive to the operator.

5 Conclusion

In this work, we proposed *learned task space control* of a redundant robot. We implemented the controller and assessed the reduction in the operator's effort to control a virtual robot. In the pilot user study with six participants, using *learned task space control* significantly reduced the effort compared to *naive task space control*.

Acknowledgments. We gratefully acknowledge funding by the Werner Siemens Foundation through the MIRACLE project, and we thank Prof. Dr. med. Niklaus F. Friederich for his continuous support with respect to medical questions. We thank the participants for their time and volunteering to take part in the study.

References

1. Siciliano, B.: Kinematic control of redundant robot manipulators: a tutorial. *J. Intell. Robot. Syst.* **3**(3), 201–212 (1990)
2. Moradi, H., Lee, S.: Joint limit analysis and elbow movement minimization for redundant manipulators using closed form method. In: *International Conference on Intelligent Computing*, pp. 423–432. Springer, Berlin (2005)
3. Flacco, F., De Luca, A., Khatib, O.: Motion control of redundant robots under joint constraints: saturation in the null space. In: *2012 IEEE International Conference on Robotics and Automation*, pp. 285–292. IEEE (2012)
4. Sandoval, J., Su, H., Vieyres, P., Poisson, G., Ferrigno, G., De Momi, E.: Collaborative framework for robot-assisted minimally invasive surgery using a 7-DoF anthropomorphic robot. *Robot. Auton. Syst.* **106**, 95–106 (2018)
5. Vigoriti, F., Ruggiero, F., Lippiello, V., Villani, L.: Tracking control of redundant manipulators with singularity-free orientation representation and null-space compliant behaviour. In: Ficuciello, F., Ruggiero, F., Finzi, A. (eds.) *Human Friendly Robotics*, pp. 15–28. Springer, Cham (2019)
6. Celebi, M.E., Celiker, F., Kingravi, H.A.: On Euclidean norm approximations. *Pattern Recogn.* **44**(2), 278–283 (2011)
7. Huynh, D.Q.: Metrics for 3D rotations: comparison and analysis. *J. Math. Imaging Vis.* **35**(2), 155–164 (2009)
8. Karnam, M., Parini, R., Eugster, M., Cattin, P., Rauter, G., Gerig, N.: An intuitive interface for null space visualization and control of redundant surgical robots. In: *Proceedings on Automation in Medical Engineering*, vol. 1. Infinite Science Publishing (2020). <https://doi.org/10.18416/AUTOMED.2020>