Control of Wall Mounting Robot

Practical Implementation and Experiments

Damgaard, Malte Rørmose; Pedersen, Rasmus; Hansen, Karl Damkjær; Bak, Thomas

Published in:
IFAC-PapersOnLine

DOI (link to publication from Publisher):
10.1016/j.ifacol.2020.12.2722

Publication date:
2020

Document Version
Publisher's PDF, also known as Version of record

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 53-2 (2020) 10025–10030

Control of Wall Mounting Robot: Practical Implementation and Experiments

Malte R. Damgaard * **Rasmus Pedersen** * **Karl D. Hansen** * **Thomas Bak** *

* *Department of Electronic Systems, Automation and Control, Aalborg University, Denmark, (e-mail: {mrd, rpe, kdh, tba}@es.aau.dk).*

Abstract: Robots are gaining traction in all industries, not only to replace manual labour but also to collaborate and enhance both human and robot skills and abilities. In this paper, we revisit the trajectory following control design for the WallMo Robot, which is a collaborative wall mounting robot used in the construction industry. The theoretical foundation for a model-free control strategy, handling actuator constraints was presented by Sloth and Pedersen (2017) and verified through simulations. In this paper, the research is extended to also include practical implementation considerations and experimental testing on a real WallMo robot. The implemented control strategy differs from what was presented earlier, but still exhibits tight trajectory tracking - solving the control problem in a practical setting.

Keywords: Robotics technology, Motion control systems, Robots in construction

1. INTRODUCTION

Traditionally, a large part of the construction industry is based on manual labour. However, as robotic technologies mature, their use in this space becomes increasingly attractive. They can, e.g., help improve the work environment by alleviating heavy lifting and consequently improve productivity. In this work, we consider the robot WallMo (2019), which is designed for mounting glass partition walls. These walls typically consist of several 2 to 3 m tall segments, with a mass of up to 150 kg. The installation of each segment requires accuracy of 1.5 mm, and instead of fully automating this complex procedure, a human operator is present to guide the robot with a joystick. To utilize the robots full productivity potential, trajectories should be executed as fast as possible, resulting in actuators, at times, reaching their torque limitations. Whenever a constraint is encountered, it is still important that the trajectory is followed tightly to give the precision mentioned above. Therefore, these constraint phenomena must be handled by the controller.

The studied control problem can be considered as a time-optimal motion planning problem, which has great importance in the control of robots because it directly affects the productivity, which consequently affects the return of investment when using a robotic solution, as explained by Constantinescu and Croft (2000). The importance is also illustrated through numerous papers addressing the theory behind time-optimal motion planning, such as von Stryk and Bulirsch (1992) and Gasparetto et al. (2015). In Verscheure et al. (2009), they show how the problem can be decomposed into two separate subproblems, where the first handle path planning and the second trajectory planning.

In many practical scenarios, torque and velocity constraints on actuators need to be taken into consideration, effectively expanding the problem to path-constrained time-optimal control, as in Chen and Desrochers (1989) and Pfeiffer and Johanni (1987). This problem is also addressed by Verscheure et al. (2009), where they cast it as a convex optimization problem, which can be effectively solved. However, this approach require considerable computational effort, and the problem increases with system degree and complexity. Another method is to exploit the dynamic model of the system and parameterize it in the path parameters. The dynamic constraints can be taken into account in the trajectory planning by dynamic scaling of the trajectory as illustrated in (Siciliano et al., 2009, Section 7.7), where violations of constraints are mitigated by scaling a given trajectory based on the dynamic model of the system along with a predefined parametrization of the scaling function.

There are also model-free approaches, such as using reinforcement learning, to solve the trajectory optimization problem Akrour et al. (2016). However, this approach relies on multiple executions of the system to gather enough data for being able to acceptably follow a given trajectory, which might not be feasible in many practical applications.

In the WallMo application, a joystick gives a reference for the time derivative of the path parameter, given a set of predefined paths. Therefore, a path does not have a specified goal-point. Additionally, the load's mass and point of attachment are unknown, as well as the dynamic parameters of the robot itself. This requires the trajectory to be generated online and adapted according to the current operating conditions, which is similar to the standard constraint handling in integrator anti-windup, which was explained by Åström and Hägglund (2006).

This paper extends the research presented by Sloth and Pedersen (2017), where a method for minimizing the execution time of a path-constrained motion was given. This

10.1016/j.ifacol.2020.12.2722

solution only used the kinematic model of the robot, given the uncertainty in robot dynamics and unknown load characteristics. Each joint has an individual velocity controller receiving reference velocities from an operational space controller taking input from the joystick. To maximize trajectory following speed in constrained operation, a switching law was proposed, with switching depending on the path parameterization. However, the approach was only validated through simulation, lacking the practical insights. In this work, the control problem is revised in a practical setting, showing that parameterization is not possible for all paths; thus an alternative approach is needed. This alternative control structure is the main contribution of this paper, along with implementation considerations and testing on a real WallMo robot.

The remainder of the paper is organized as follows: Section 2 revisits the original problem formulation and sketches how a glass panel can be installed with a robot; subsequently, the alternative control structure is given in Section 3. This is followed by simulation studies in Section 4 and experiments in Section 5. Finally, conclusions are drawn in Section 6.

2. PROBLEM FORMULATION

In this section the original problem, presented by Sloth and Pedersen (2017), of maximizing motion speed for a path-constrained robot, with unknown dynamics and actuator constraints is revisited, and the caveats of the proposed control strategy are highlighted. The focus is on the installation procedure of the glass, as this is by far the most time consuming and complex task.

The segments of the glass partition walls are mounted in two U-shaped profiles; one in the ceiling and one on the floor. The glass panel is first mated to the top bracket and then lowered into the bottom bracket. Three motion primitives were defined for this installation procedure Sloth and Pedersen (2017) and illustrated in Fig. 1 for clarity. The three motions are: *Move*, *Slide*, and *Lift*.

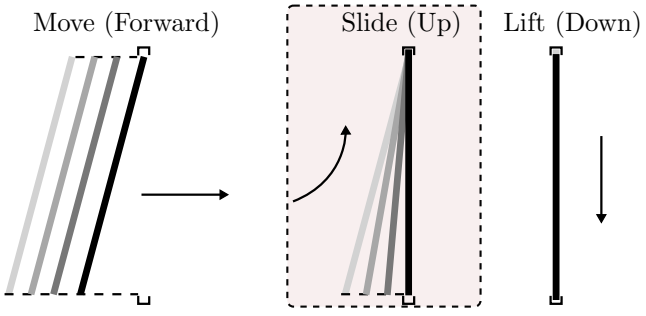

Fig. 1. A three step installation procedure for mounting a glass panel. Each movement starts at the brightest color and ends at the darkest. The motion primitive *Slide* is highlighted as it constitutes an issue when implementing the original control structure on the real robot. Source Sloth and Pedersen (2017)

To link these motion primitives to robot movements a sketch of the WallMo robot along with the definition of local joint frames can be seen in Fig. 2 and parameter values can be found in the Appendix. Notice, that for the robot to perform the three motion primitives only three joints are actively used (q_1,q_2,q_3), where the other joints are used for glass pickup and to get the glass into installation position. Therefore, the joints (q_4,q_5) are not considered further. Further, the tool frame is placed at the top of the glass panel.

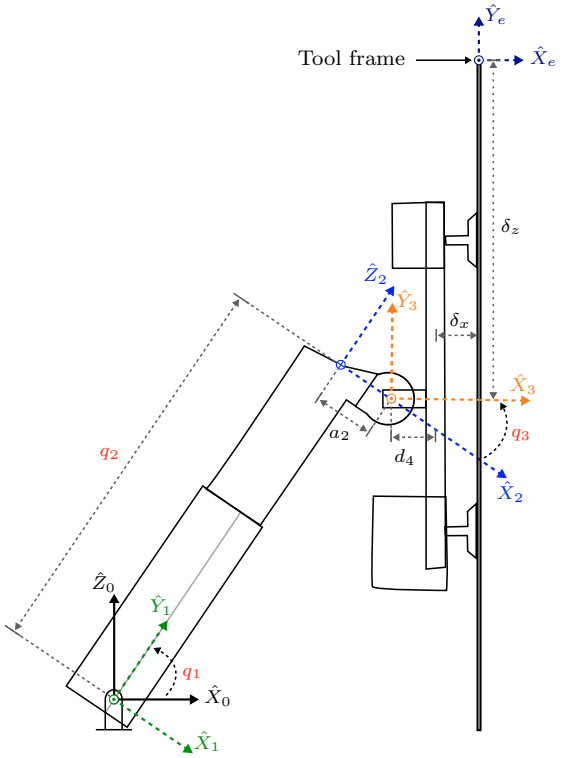

Fig. 2. Sketch of the Wallmo robot along with local joint frames. The generalized coordinates used in the installation procedure (q_1,q_2,q_3) are marked by red text color. The DH parameters can be found in the Appendix. Source Sloth and Pedersen (2017)

Let the tool frame configuration (given in the base frame) be denoted by

$$x_e = (x,y,\theta), \tag{1}$$

where $x \in \mathbb{R}$ is the distance in the x-axis from base frame to tool frame, $y \in \mathbb{R}$ is the distance in the y-axis from base frame to tool frame, and $\theta \in \mathbb{R}$ is the angle about the base frame $\hat{Y}_0$ axis. Now let $x_{e,0} = (x_0,y_0,\theta_0)$ be the initial configuration of the tool frame. Then paths for the three motion primitives can be formulated as follows, where each motion is parameterized in the path parameter $s \in \mathbb{R}$

$$\begin{aligned}
\beta_{\text{Move}} &: (s,x_{e,0}) \mapsto (x_0 + s, y_0, \theta_0) \\
\beta_{\text{Slide}} &: (s,x_{e,0}) \mapsto (x_0, y_0 + 2\delta_y(\cos(\theta_0 + s) - \cos(\theta_0)), \theta_0 + s) \\
\beta_{\text{Lift}} &: (s,x_{e,0}) \mapsto (x_0, y_0 + s, \theta_0)\,,
\end{aligned}$$

The robot is controlled with a joystick, which outputs a reference speed for the selected motion, $\dot{s} = \alpha\bar{\dot{s}}$, where $\alpha \in [-1,1]$ and $\bar{\dot{s}}$ is an upper limit for the velocity reference that is selected e.g. based on safety constraints, and each joint has an associated velocity PID controller. In Sloth and Pedersen (2017) the nominal control structure in Fig. 3 was proposed, where the joystick output was mapped into reference signals for an operational space controller, through the $\beta(\cdot)$ functions above. Furthermore, to ensure path following when actuator saturation occurs

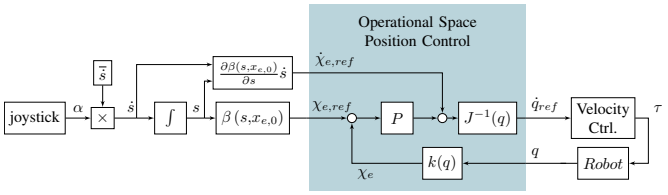

Fig. 3. The nominal control structure presented by Sloth and Pedersen (2017).

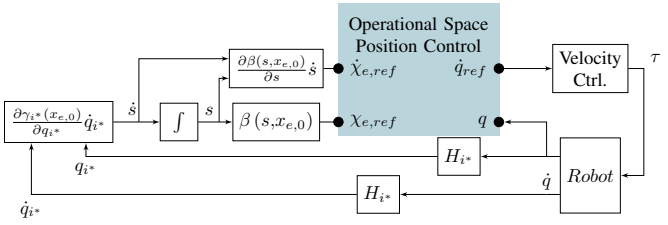

Fig. 4. The control structure presented by Sloth and Pedersen (2017) for the case in which one or more motors are saturated. The expression for H_{i^*} can be found in the original paper.

the control structure in Fig. 4 was proposed, where a set of, $\gamma(\cdot)$, functions and their derivatives, $\dot{\gamma}(\cdot)$, for each motion primitive needs to be computed. In Sloth and Pedersen (2017) the calculations of these, $\gamma(\cdot)$, $\dot{\gamma}(\cdot)$, functions was shown for the motion primitive *lift* and a procedure that ensures path following in the case of actuator saturation was given.

While the procedure is mathematically correct, an issue occurs when trying to compute the, $\gamma(\cdot)$, functions for the motion primitive *Slide*. It turns out that deriving these functions for the motion primitive *Slide* is too convoluted, if not impossible, for the proposed control approach to be useful in a practical setting. The issues stems from substituting the β_{Slide} expression into the inverse kinematics and then solving this expression for path parameter s. The inverse kinematics is given in Sloth and Pedersen (2017), but restated in Eq. 2 for completeness.

$$\begin{aligned} q_2 &= \sqrt{\tilde{x}^2 + \tilde{y}^2 - a_2^2} \\ q_1 &= \text{asin}\left(\frac{q_2\tilde{y} + a_2\tilde{x}}{q_2^2 + a_2^2}\right) \\ q_3 &= \frac{\pi}{2} + \theta - q_1 \end{aligned} \tag{2}$$

where

$$\begin{aligned} \tilde{x} &= x - \delta_y \text{cos}\left(\frac{\pi}{2} + \theta\right) - \delta_x \text{sin}\left(\frac{\pi}{2} + \theta\right) \\ \tilde{x} &= x - \delta_y \text{sin}\left(\frac{\pi}{2} + \theta\right) + \delta_x \text{cos}\left(\frac{\pi}{2} + \theta\right) \end{aligned}$$

This procedure results in huge expressions, which it have not been possible to solve even with tools such as the MATLAB symbolic toolbox MathWorks (2019).

The remainder of this paper addresses this problem.

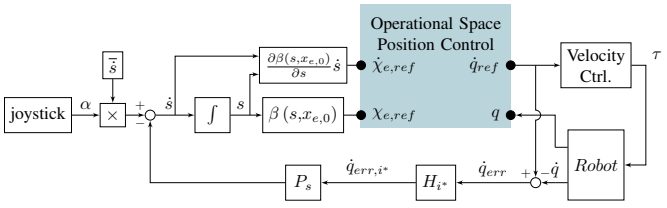

Fig. 5. New proposed control structure, where actuator saturation is handled through classical feedback control.

3. NEW CONTROL STRUCTURE

The purpose of this section is to present an alternative control structure for the motion primitive *Slide*. Although, the control structure presented in Sec. 2 is mathematically correct, it was not possible to derive the required functions, $\gamma_{\text{slide},i}$ for i = 1,2,3, needed to use this control structure together with the motion primitive *Slide*. Basically, the control structure presented in Section 2 limits the reference generated by the user's interaction with the joystick, $\dot{s}$, to the largest value allowed by the saturated motor, $\bar{\dot{s}}_{i^*}(s)$, and the chosen motion primitive. Based on this observation, the control structure illustrated in Fig. 5 is proposed. This control structure mimics this behaviour, but through more classical feedback control mechanisms.

The functionality of the control structure illustrated in Fig. 5 is as follows. Whenever, one of the motors of the robot are saturated the reference signal, $\dot{s}$, is reduced proportionally to the joint velocity error caused by the saturated motor, $\dot{q}_{err,i^*}$.

It should be noted that in general

$$\bar{\dot{s}}\alpha - P_s\dot{q}_{err,i^*} \neq \frac{\partial\gamma_{i^*}(x_{e,0})}{\partial q_{i^*}}\dot{q}_{i^*} = \bar{\dot{s}}_{i^*}(s) \tag{3}$$

for a constant gain, P_s. Therefore, one should be careful when choosing the gain P_s, since the feedback have to lower $\dot{s}$ enough to ensure that the motor with index i gets out of saturation. In theory, for this to be true the inequality

$$\left|\bar{\dot{s}}\alpha - P_s\dot{q}_{err,i^*}\right| \leq \left|\bar{\dot{s}}_{i^*}(s)\right| \tag{4}$$

should be true whenever the i'th motor saturates. On the other side if P_s is chosen too large the system would potentially become unstable. Notice that a thorough stability analysis is outside the scope of this paper and is therefore left for future work.

4. SIMULATION

To further investigate which gains, P_s, are appropriate for the new control structure proposed in Section 3 and also to substantiate the functionality of this new control structure, we present a simulation study.

To investigate the performance of the new control structure we use the kinematic and dynamic simulation model of the robot that was presented by Sloth and Pedersen (2017) and implement the full control structure, as illustrated in Fig. 3 and Fig. 5. For comparison, the old

control structure, as illustrated in Fig. 3 and Fig. 4, is also implemented in Simulink. The different controller gains are shown in Table 2.

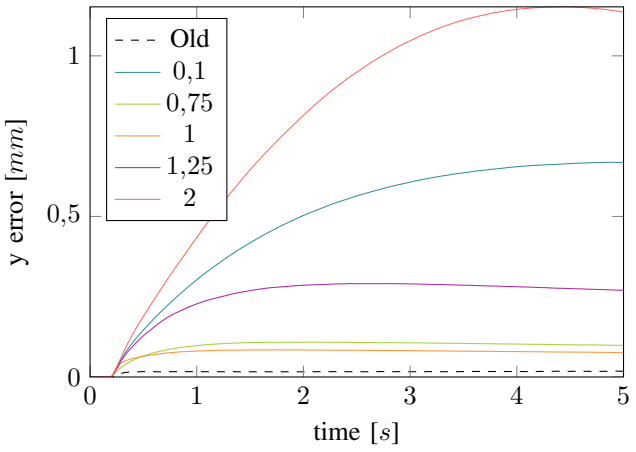

Fig. 6. Plot showing the influence of different control gains, P_s, on the tracking error of the main variable for the motion primitive *Lift*. For comparison the result of a simulation for the old control structure is also shown. The simulation shows that $P_s \approx 1$ seems appropriate with the Operational Space Position Control gains given in Table 2.

To compare the performance of the new and the old control structure, simulations are first performed for the motion primitives *Move* and *Lift* for which it is possible to implement both control structures. Fig. 6 shows the results of simulations for the motion primitive *Lift* with different controller gains of the new control structure and for comparison also the result of a simulation with the old control structure presented in Sec. 2. Fig. 6 shows that the new control structure with an appropriate gain can perform nearly as well as the old control structure and that it in simulations performs more than adequate for this application.

Another important result from the simulation study, that is not evident from Fig. 6, is that low gains of the operational space position controller and velocity controller have a rather large impact on the performance of the overall system performance. Since both of the control structures are cascaded control structures, this is expected. However, the control structure presented in Sec. 3 seems to be more affected by this than the control structure shown in Sec. 2. In an ideal simulation environment, this is not a problem, but in the real systems, this could become a problem due to unmodeled dynamics and/or noise.

Although the new control structure shows more tracking error, it is, in fact, implementable. Fig. 7 shows data of the tool frame configuration, x_e, for a simulation of the new control structure and the motion primitive *Slide*.

As seen from Fig. 7 the robot tracks the references for the *Slide* motion primitive even when one of the motors are in saturation, so we will conclude that the new control structure seems adequate for this application provided that the simulation model presented by Sloth and Pedersen (2017) matches the dynamics of the real plant.

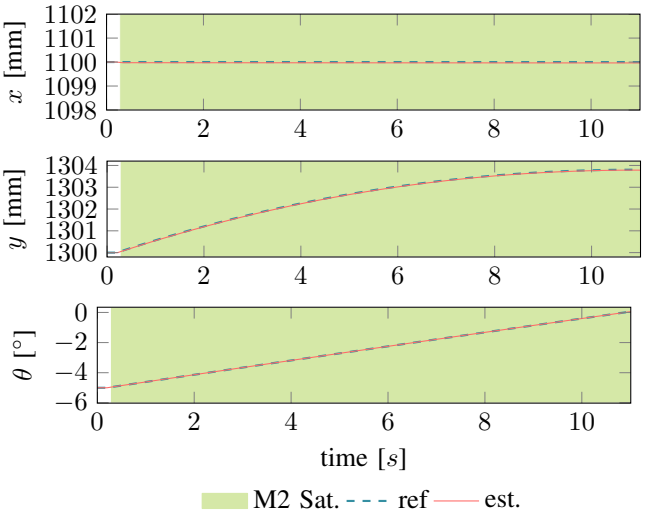

Fig. 7. Plot of simulated data for the new control structure with $P_s = 1$ and the motion primitive *Slide*. The region of the plots highlighted with green is the time instances at which one of the motors where in saturation.

5. EXPERIMENTAL TEST

The purpose of this section is to present the performance of the two control structures described in Section 2 and Section 3 evaluated through experiments. To verify the performance of the two control structures a full-size robot was developed in the WallMoBot Project. The robot is equipped with appropriate motors, motor drivers, encoders for measuring the position of the joints, other necessary electronics, and a cRIO-9039 Controller from National Instruments. The control algorithm was implemented in National Instruments Labview and loaded onto the cRIO.

The simulation model used in Sec. 4 is based on a model of the robot with torque as input. However, the motors on the developed robot are actually controlled by a PWM input signal, why each of the controllers gains was re-tuned on the robot. The gains of the re-tuned controllers can be seen in Table 3. Besides re-tuning the controller gains, it was decided to implement a feature that bypasses all the controllers forcing the motors to stop when the user stops giving inputs with the joystick. This was decided since practical tests of the system showed that anything else would seem unnatural, as the operator wanted the robot to stop moving when they let go of the joystick. But without the feature, the controller would try to minimize any tracking error, making the robot do undesirable movements after the operator wanted it to stop.

To obtain adequate measurements, a Vicon motion capture system was used as ground truth. As shown on Fig. 8, Vicon markers were placed on a glass panel with known dimensions in such a way that the configuration of the tool frame, x_e, could be tracked. With the glass attached to the robot, a series of movements with the motion primitives were performed. Fig. 9 and 10 shows the measurements obtained for the motion primitives *Lift* and *Slide* which represents the performance of the original and new control structure respectively.

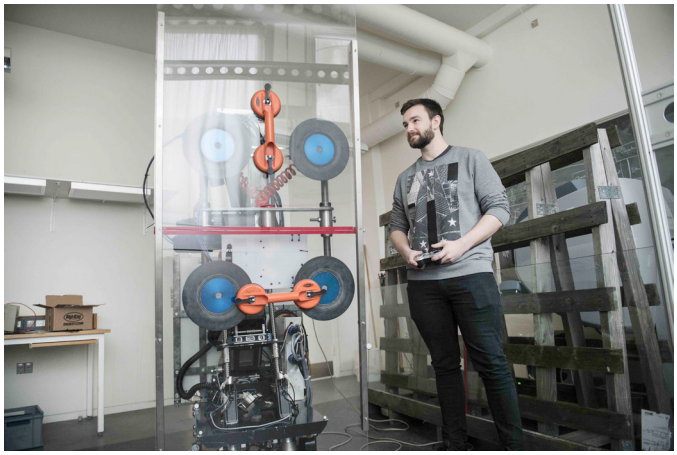

Fig. 8. Physical robot lifting a glas panel with Vicon markers, for measuring the performance of the control structures.

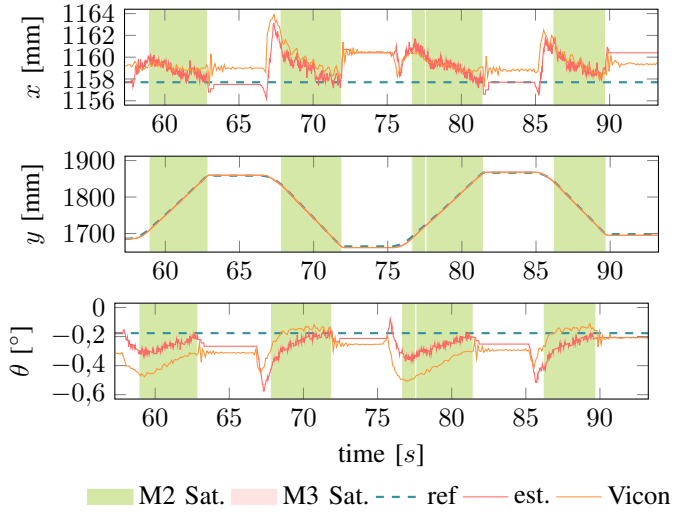

Fig. 9. Plot of data gathered while using the motion primitive *Lift*. The regions of the plots highlighted with green and red are the time instances at which one of the motors was in saturation.

From Fig. 9 it is seen that the original control structure first presented by Sloth and Pedersen (2017) and summarized in Section 3 in general is able to track the references of the operational space position controller closely even when the motors are saturated. However, at $t \approx 67s$, $t \approx 72s$, $t \approx 85s$ and $t \approx 90s$, the x variable makes a sudden jump away from its reference. This happens approximately when the y-velocity changes direction, and thus the phenomena could be due to uncompensated coulomb friction. Another possibility could be that the robot is not perfectly rigid combined with the implemented feature bypassing all the controllers when the user stops giving input to the system. As such, this is not caused by the original control structure and it must be concluded that the original control structure effectively handles actuator constraints not only in simulation but also in a practical system.

From Fig. 10 one thing is noticeable. At some time instances, there seems to be a relatively large deviation of approx. 10 mm between the Vicon system and the

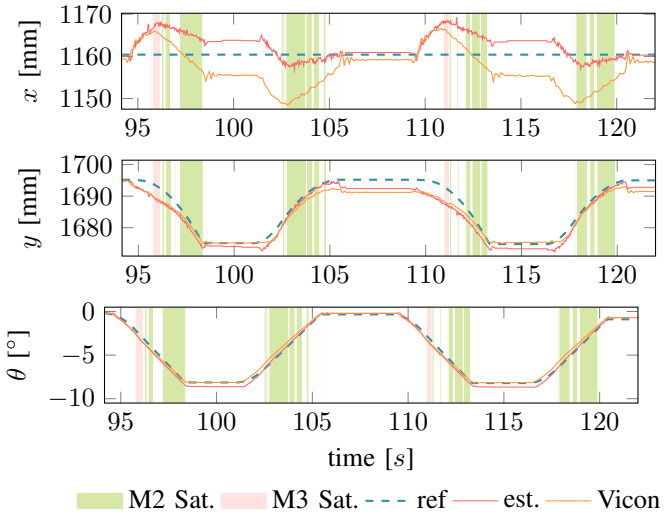

Fig. 10. Plot of data gathered while using the motion primitive *Slide*. The regions of the plots highlighted with green and red are the time instances at which one of the motors where in saturation.

robots own estimate of the x variable. This is believed to be due to the robot not being totally rigid, and since the robots own estimate is only based on local sensors measuring the position of each joint. This is a mechanical problem that cannot be handled in the control structure unless the robot's estimate of the configuration of the window frame is improved. This will have to be corrected in the next iteration of the mechanical design of the robot. Disregarding this mechanical design problem, and by comparing the robots own estimate of the configuration of the tool frame, x_e, with its reference, it is seen that the new control structure is actually performing nearly as well as the original control structure.

Besides the precision test discussed above some practical test with a potential end-user of the robot has been performed. These tests have demonstrated that both of the control structures are more than adequate for the needed application and that any minor tracking error that these control structures cannot handle is effectively compensated for by the human in the loop. Finally, these practical tests have shown that the developed robot is indeed capable of assisting humans in installing large and heavy glass panels; both cutting the needed number of craftsmen to do the installation down to a single operator, but also freeing this operator from doing any heavy lifting.

6. CONCLUSION

In this paper, the original trajectory following approach of the wall mounting robot WallMo was revisited. The computational shortcomings of this approach, making full implementation on a real robot infeasible, was highlighted, and an alternative control structure was proposed. This approach relied on classical feedback control to circumvent the computation issues inherent to the original control structure. The performance differences between the two controllers were illustrated through simulation studies, showing insignificant differences. Furthermore, the new controller was fully implemented on a WallMo robot and validated through experimental testing.

ACKNOWLEDGEMENT

This work is supported by Innovation Fund Denmark in project number 5150-00007A called WallMoBot.

APPENDIX

This appendix provides the kinematic model parameters of the robot (Table 1) and the controller parameters (Tables 2 and 3) used in this paper. Notice that the large gain differences between the control parameters used in simulations and on the actual robot is due to the actual robot is PWM controller instead of torque controlled as assumed in the simulation model.

Table 1. Kinematic DH Parameters

i	a_{i-1} [m]	d_i [m]	θ_i [rad]	α_{i-1} [rad]
1	0	0	$q_1 - \pi/2$	$\pi/2$
2	0	q_2	0	-$\pi/2$
3	a_2	0	$q_3 - \pi/2$	$\pi/2$
4	0	d_4	$q_4 + \pi/2$	$-\pi/2$
5	0	0	q_5	$\pi/2$

Table 2. Control parameters Used in simulations.

Controller	Parameter	Value
Vel. PID Joint 1, 2	P, I, D gains	5e5, 5e8, 5e3
Vel. PID Joint 3	P, I, D gains	5e5, 5e8, 5e2
Operational Space	P	diag(0,5; 0,05; 0,1)

Table 3. Re-tuned control parameters used on the actual robot.

Controller	Parameter	Value
Velocity PID Joint 1	P gain, I gain	0,5 and 6
Velocity PID Joint 2	P gain, I gain	6,75 and 50
Velocity PID Joint 3	P gain, I gain	1,5 and 6
Operational Space	P	diag(0,5; 0,5; 0,5)

The top of the glass panel is considered the origin of the Tool Frame (Frame $\{e\}$). The transformation from Frame $\{5\}$ to Frame $\{e\}$ is given by the transformation:

$$ {}^{5}_{e}T = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & \delta_x \\ 0 & 0 & -1 & -\delta_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5} $$

where δ_x is the offset distance of the glass panel from the final joint of the robot, and δ_z is the height of the glass panel from the attachment point to the top. The tool frame is aligned with the base frame in the default configuration ($q = 0$).

REFERENCES

Akrour, R., Abdolmaleki, A., Abdulsamad, H., and Neumann, G. (2016). Model-free trajectory optimization for reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*.

Åström, K.J. and Hägglund, T. (2006). *Advanced PID Control*. International Society of Automation (ISA).

Chen, Y. and Desrochers, A.A. (1989). Structure of minimum-time control law for robotic manipulators with constrained paths. In *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, 971–976 vol.2.

Constantinescu, D. and Croft, E.A. (2000). Smooth and time-optimal trajectory planning for industrial manipulators along specified paths. *Journal of Robotic Systems*, 17(5), 233–249.

Gasparetto, A., Boscariol, P., Lanzutti, A., and Vidoni, R. (2015). *Path Planning and Trajectory Planning Algorithms: A General Overview*, 3–27. Springer International Publishing.

MathWorks (2019). Symbolic Math Toolbox for MATLAB. `https://mathworks.com/products/symbolic.html`. Accessed: 2019-08-05.

Pfeiffer, F. and Johanni, R. (1987). A concept for manipulator trajectory planning. *IEEE Journal on Robotics and Automation*, 3(2), 115–123.

Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G. (2009). *Robotics: Modelling, Planning and Control*. Springer-Verlag London.

Sloth, C. and Pedersen, R. (2017). Control of wall mounting robot. In *Proceedings of the 2017 IFAC World Congress*.

Verscheure, D., Demeulenaere, B., Swevers, J., Schutter, J.D., and Diehl, M. (2009). Time-optimal path tracking for robots: A convex optimization approach. *IEEE Transactions on Automatic Control*, 54(10), 2318–2327.

von Stryk, O. and Bulirsch, R. (1992). Direct and indirect methods for trajectory optimization. *Annals of Operations Research*, 37(1), 357–373.

WallMo (2019). The panel mounting co-bot. URL `https://www.wallmo.dk`. Accessed: 2019-08-05.