Autonomous Mobile 3D Printing of Large-Scale Trajectories

Julius Sustarevas[1], Dimitrios Kanoulas[1], Simon Julier[1]

***Abstract*— Mobile 3D Printing (M3DP), using printing-in-motion, is a powerful paradigm for automated construction. A mobile robot, equipped with its own power, materials and an arm-mounted extruder, simultaneously navigates and creates its environment. Such systems can be highly scalable, parallelizable and flexible. However, planning and controlling the motion of the arm and base at the same time is challenging and most deployments either avoid robot-base motion entirely or use human prescribed robot-base paths. In a previous paper, we developed a high-level planning algorithm to automate M3DP given a print task. The generated robot-base paths avoid collisions and maintain task reachability. In this paper, we extend this work to robot control. We develop and compare three different ways to integrate the long-duration planned path with a short horizon Model Predictive Controller. Experiments are carried out via a new M3DP system — Armstone. We evaluate and demonstrate our algorithm in a 250 m long multi-layer print which is about 5 times longer than any previous physical printing-in-motion system.**

I. Introduction

The sustainability goals of the 21st century are driving the need to automate the building and construction industry [1], [2]. Whilst a variety of digital robotic manufacturing solutions have already been widely adopted for off-site component prefabrication [3], [4], in-situ automation remains mostly limited to site inspection and monitoring [5], [6]. However, a major exception is the use of large-scale Additive Manufacturing (AM) – also known as 3D printing – systems. AM methods have demonstrated increased efficiency and waste reduction [7]–[9], however sustainable deployment and material use challenges are still open [10]. Despite this, such systems are already being commercialised [11]–[13] and deployed to tackle sustainability and housing crises [14].

Most existing AM construction solutions use static gantries or arms. They are highly restrictive and costly to deploy [8]. An alternative is a Mobile 3D Printing (M3DP) system, shown in Fig. 1, which uses smaller mobile manipulator robots. Such systems have unlimited planar range, high agility to track more complex print trajectories [15] as well as potential to parallelise the printing task [16].

Despite their advantages, there are very few works that explore M3DP applications [16]–[22]. Autonomously tracking print paths, which are much larger than the robot, requires a robust motion and path planning approach. This remains challenging and, although these works demonstrate the feasibility of M3DP, they often make restricting assumptions such as human-prescribed robot-base paths or decoupled robot-base and arm control. In turn, this reduces system autonomy and agility. Consequently, to date there has been no demonstration of an M3DP system that can autonomously plan and 3D print a structure given just a print task.

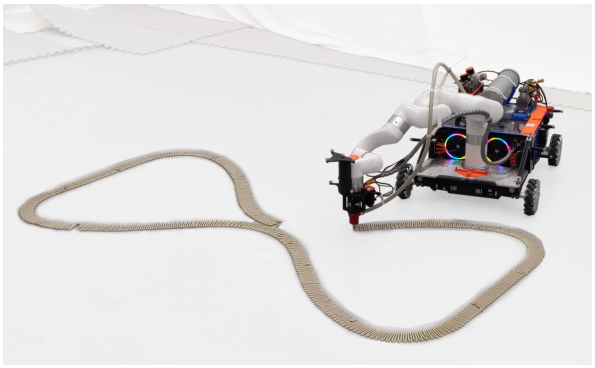


Fig. 1: The Armstone autonomous mobile 3D printing system.

In this paper, we extend our previous work on Task Consistent Path Planning (TCPP) to hardware by using TCPP-derived robot-base paths to complement a whole-body Model Predictive Controller (MPC). Our approach is, thus, globally informed and benefits from locally optimal combined base-arm motion planning. We compare three ways in which the robot-base references can be added to the MPC formulation. Finally, we demonstrate printing of long (250 m+) trajectories, over multiple layers, via a purpose-built mobile manipulator, named Armstone.

The remainder of this work is organised as follows. The next section overviews related work in mobile 3D printing and MPC. Sec. III briefly covers the Armstone robot hardware design and the MPC setup used in this work. Sec. IV presents how we integrated robot-base reference trajectories into the MPC formulation. Next, Sec. V and VI show large-scale printing experiments. Lastly, a discussion of results and future work can be found in Sec. VII.

II. Related Work

A. The Challenges of Mobile 3D Printing

Conceptually, the feasibility of Mobile 3D Printing has been demonstrated. In our earlier work [17], we showed that tracing print trajectories, from a moving mobile manipulator, results in a marginal end-effector error increase, compared to a stationary approach. Another work, by Tiryaki et al. [19], realised the first concrete printing mobile manipulator system. However, both of these works used a manually-specified robot-base path to keep the printing task within the reach of the robot.

[1]Department of Computer Science, University College London, Gower Street, WC1E 6BT, UK. julius.sustarevas@cs.ucl.ac.uk

This work was partially supported by the UKRI Future Leaders Fellowship [MR/V025333/1] (RoboHike). Optitrack tracking system has been provided by UKRI AABM Research Grant [EP/N018494/1].

This robot-base path finding problem was tackled in our work on Task-Consistent Path planning (TCPP) [18]. We proposed a methodology to compute the robot-base path from the desired print trajectory and the knowledge of robot kinematics. The TCPP algorithm took into account M3DP-specific challenges, such as evolving obstacles created by the printing process and manipulator reachability preservation.

However, these existing efforts decouple the robot-base and end-effector control. The local manipulator trajectory is obtained by subtracting the print path from the print task. In that way, the error that was generated by following the prescribed robot-base trajectory was directly translated to the end-effector without any mitigation by the manipulator.

B. Path Planning for M3DP

Early research exploring the path planning problem for mobile manipulators with given end-effector paths, tended to treat the robot systems as redundant manipulators and used sampling-based planners, such as PRM or RRT [23]–[27]. In those planners, the task-constraint was inverted and a path was formed from randomly sampled admissible robot configurations [28], [29]. To help reduce the dimensionality of the problem, Nagatani et al. [30] proposed planning the robot-base path separately, while assuring that the task remains in a manipulable region in the local robot arm frame.

We used Nagatani et al.'s approach to develop our TCPP algorithm specifically for mobile 3D printing [18]. We used the Inverse Reachability Map (IRM) [31] to help sample and validate robot-base poses along a print path during the RRT^* search. Given a robot-base pose and an associated point on the print path, the IRM provides a Reachability Index (RI) for the print point. RI is a metric [32], [33] associated with a voxel in a manipulator's local workspace. For many $SE(3)$ poses inside a voxel, it is the fraction of poses with existing Inverse Kinematics (IK) solutions. The IK computation takes into account both the robot self-collisions and its joint limits. Intuitively, RI relates to robots ability to position its end-effector at a given point. Thus, by imposing a lower bound on RI when validating robot-base poses, we were able to find robot-base paths for long print trajectories, while assuring the print task remains well reachable by the arm.

C. Control for M3DP

Given the robot-base and end-effector paths, an arm joint trajectory can be computed. However, following both robot-base and arm trajectories still requires coordination and robustness. Tiryaki et al. [19] used independent robot-base and arm controllers. The base used a Model Predictive Controller following a prescribed path with respect to an on-board localisation system. The arm used a low-level joint trajectory controller. This meant that the end-effector error was entirely reliant on the robot-base trajectory tracking error. However, as 3D printing is a relatively slow process, M3DP requires slow and steady robot-base velocities. This is often difficult to achieve due to breakaway friction, torque requirements, or uneven terrain. A similar problem was also identified in [22]. Using a separate robot-base and arm controllers led to increases in end-effector error, especially at the start of the print trajectories as the robot-base struggled to render accurate motion from full stop.

Recently, there has been a great deal of research interest in whole-body motion control for various quadruped, mobile manipulator, and humanoid robots [34]–[37]. Specifically, Pankert et al. [38] proposed a control strategy for continuous process tasks. They used a Sequential Linear Quadratic Model Predictive Control (SLQ-MPC) formulation to demonstrate end-effector trajectory planning, admittance control, and collision avoidance. The SLQ-MPC controller rolls-out the current control policy over a prediction horizon T, linearizes the system about the resulting states, and minimises a cost function L integral over T:

$$\begin{aligned} J(\mathbf{x},\hat{\mathbf{x}},\mathbf{u}) &= \int_{t_0}^{t_0+T} L(\mathbf{x}(\tau),\hat{\mathbf{x}}(\tau),\mathbf{u}(\tau))d\tau, \\ L(\mathbf{x},\hat{\mathbf{x}},\mathbf{u}) &= \sum C_i + \sum B_i + \mathbf{u}^\intercal \mathbf{R}\mathbf{u} \end{aligned} \tag{1}$$

Here, $\mathbf{x}$ is the system state, $\hat{\mathbf{x}}$ is the reference trajectory, and $\mathbf{u}$ are the inputs. L consists of cost terms C_i, control barrier terms B_i, and the input penalty term $\mathbf{u}^\intercal\mathbf{R}\mathbf{u}$. Pankert et al. [38] used a quadratic end-effector tracking cost term and imposed soft inequality constraints, using B_i, for manipulator joint position and velocity limits, as well as the collision avoidance and admittance control features. The computed policies were evaluated using the state estimates while the inputs were passed to the motor controllers. For further details, e.g. derivative approximation, see [38].

In Pankert et al. [38], a failure case occurred when the collision avoidance and end-effector tracking objectives were competing. Such cases might occur often in 3D printing applications. Print paths are vastly longer than the predictive horizon and, thus, obstacles being created by printing can prevent the robot from completing the task. To illustrate this, consider the scenario in Fig. 2. The robot prints, left to right, a trajectory that carries on beyond immediate reach. The predictive horizon fails to see the need for the robot-base to move rightwards before such motion would cause a collision or compromise end-effector tracing accuracy.

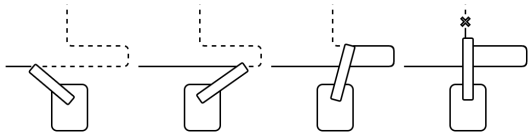


Fig. 2: Trajectories substantially longer than predictive horizon may lead to infeasible states.

Therefore, in this work, we supplement the SLQ-MPC controller with a TCPP-derived robot-base reference trajectory to provide foresight and allow tracking very long trajectories. We use an open-source implementation for SLQ-MPC [38]. This is based on the OCS2 [36], [39] library for optimisation, RobCoGen [40] for kinematics and CppAD for auto differentiation. The formulation of costs and constraints used is presented in Sec. III-B and IV. We next describe the Armstone robot we developed to carry out our M3DP experiments.

III. Armstone M3DP System

In this section, we present the development of a new robot, named Armstone, which is shown in Fig. 3. Armstone is a low-cost mobile manipulator designed for M3DP. It is a research platform and is not intended for realistic deployment. It is material and power independent, allowing it to leverage the advantages of mobile manipulators for AM applications. Here, we cover hardware and software components as well as present how the robot is modelled by the SLQ-MPC controller.

A. Robot Hardware

The robot-base platform has a $0.8\,\text{m} \times 0.62\,\text{m} \times 0.3\,\text{m}$ footprint, uses Mecanum wheels for holonomic motion and weighs approximately 20 kg. The robot-base is powered by a $48\,\text{V}, 20\,\text{A h}$ battery and uses four Maxon EC45 70 W motors with 91:1 reduction gearbox. The motors are run using VESC motor controllers. Armstone features a low-cost, 6DoF, 5 kg payload, 0.7 m reach uFactory xArm manipulator. The manipulator is mounted at the bottom of the base to maximise reach at ground level.

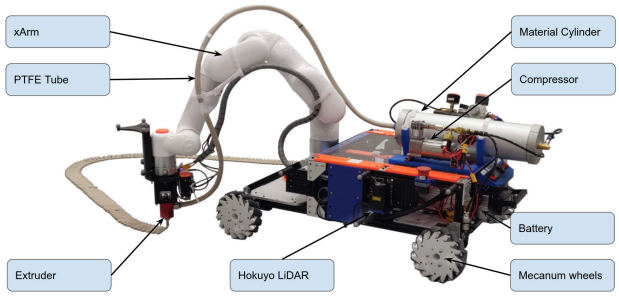


Fig. 3: Armstone robot's major components.

The robot carries two Intel Nuc (2-cores and 4-cores @4 GHz) computers as well as the xArm DC control box. Internal communications pass through a 100Mbps network switch and the robot hosts a wireless network through an on-board router.

B. SLQ-MPC Setup

The system model closely follows that of Pankert et al. [38]. The robot state $\mathbf{x}$ consists of the manipulator joint values $\mathbf{x_a} = (\phi_1, .., \phi_6)$ and the robot-base pose, $\mathbf{x_b} \in SE(3)$, expressed as a position and a quaternion in the world frame. The robot input $\mathbf{u}$ is composed of the robot-base translational velocities v_x, v_y (in the robot-base frame), robot-base rotational velocity ω_b, and manipulator joint velocities.

$$\begin{aligned} \mathbf{x} &= [\mathbf{x_b}, \mathbf{x_a}]^\intercal \\ \mathbf{u} &= [v_x, v_y, \omega_b, \dot{\phi}_1, .., \dot{\phi}_6]^\intercal \end{aligned} \tag{2}$$

For a given task trajectory $\hat{\mathbf{p}}$, the end-effector tracking cost function C_{ee} is defined as follows:

$$\begin{aligned} C_{ee}(t, \mathbf{x}, \hat{\mathbf{x}}) &= [\mathbf{e}_{xyz}, \mathbf{e}_{rpy}]^\intercal \mathbf{Q}_{ee} [\mathbf{e}_{xyz}, \mathbf{e}_{rpy}] \\ \mathbf{e}_{xyz}(\mathbf{x}, \hat{\mathbf{p}}_{xyz}) &= FK(\mathbf{x})_{xyz} - \hat{\mathbf{p}} \\ \mathbf{e}_{rpy}(\mathbf{x}, \hat{\mathbf{p}}_q) &= FK(\mathbf{x})_q \ominus \hat{\mathbf{p}}_q \end{aligned} \tag{3}$$

where, $\hat{\mathbf{p}}$ is a part of reference trajectory $\hat{\mathbf{x}}$ and is composed of positions and quaternion orientations prescribing the print path. The C_{ee} is computed as the weighted sum of squared translational and rotational end-effector pose (as provided by forward kinematics) errors $\mathbf{e}_{xyz}$ and $\mathbf{e}_{rpy}$. The quaternion distance error is computed as in [41]. Q_{ee} is a diagonal matrix of weights.

State position and velocity limits are implemented as soft constraints by penalising constraint penetration $z_i \geq 0$ via a relaxed barrier function [42]:

$$B_i(z_i) = \begin{cases} -\mu \ln(z_i), & z_i > \delta \\ \mu \beta(z_i, \delta), & z_i \leq \delta \end{cases} \tag{4}$$

where, β is a quadratic function such that B is twice differentially continuous, and μ and δ are tuning parameters. In this work, $\mu = 5 \cdot 10^{-3}$ and $\delta = 10^{-4}$. We do not use any admittance control or collision avoidance constraints present in Pankert et al. [38].

In this work, the prediction horizon T, see Eq. 1, is always 4 s. The MPC runs at 30 Hz and resulting policies are evaluated at 100 Hz.

C. Localisation and Control

Localisation is achieved either using an external OptiTrack system or two Hokuyo 04lx LiDAR sensors mounted on the sides of the Armstone. Scans of these sensors are merged using the `ira-laser-tools` package [43] and the output used by the `slam-toolbox` localisation package [44].

The Armstone base runs a holonomic controller at 50 Hz, translating robot-base velocities v_x, v_y, ω_b to wheel angular velocities. The xArm manipulator accepts desired joint velocity inputs at 100 Hz. Robot-base and arm controllers are combined using ROS Control [45] hardware interface API.

D. Extrusion System

The Armstone's on-board printing system extends our previous work [22]. It consists of an on-board compressor pressuring a 3 L material cylinder. The pressure drives material from the cylinder to the extruder via a PTFE tube which has been heat treated to aid bending along with manipulator motion. A Stoneflower3D 6 mm nozzle extruder is mounted to the xArm flange and is controlled by one of the on-board computers. The material we use is porcelain with paper fibres as we found it most reliably travels through the feeding tube.

IV. Base-path informed SLQ-MPC

Mobile 3D Printing requires the robot to accurately follow a long end-effector trajectory. However, M3DP does not dictate a precise robot-base pose throughout printing. As long as the robot-base navigation supports the dynamic nature of the evolving environment and maintains print path reachability, the print task can be achieved successfully. Therefore, in this section we describe how a SLQ-MPC motion controller can be globally guided by a robot-base path reference that assures feasibility to complete a long print task.

A. Task-Consistent Path Planning

In our previous work, we describe how the robot-base path, can be obtained from the *RRT**-based Task-Consistent Path Planner (TCPP). The result is an ordered list of tuples $b_i = (\mathbf{x}_{b,i}, s_i)$, where $\mathbf{x}_{b,i} = (x_i, y_i, \theta_i) \in SE(2)$ and s_i is a process variable corresponding to the length along the print path line integral. The validation routine, $isValid(b_i)$, uses an Inverse Reachability Map to assure that the print path at s_i is reachable from base pose $\mathbf{x}_{b,i}$, while accounting for obstacles created by printing up to s_i. For details, please see [18].

Our original implementation did not model dynamic constraints and, as a result, the computed base paths were often jagged. However, smoother paths are preferred. In this work, we introduce the following post processing steps. Firstly, after the TCPP algorithm reaches the goal and produces a robot-base path, we continue running the *RRT** loop until the size of the underlying graph has doubled. This increases the number of available samples and provides the opportunity for the *RRT** rewire function to shorten the path. Next, we iterate through b_i in a random order and attempt to bypass unnecessary robot-base poses [46]. Namely, b_i is removed and b_{i-1} is connected directly to b_{i+1}, if densely sampled nodes along this connection all satisfy the validation function. Finally, we relax the robot-base poses as follows. We iterate through b_i, except for the first and last nodes, in a random order and sample m random tuples $b_i^* = (\mathbf{x}_{b,i}^*, s_i^*)$ in a small radius about b_i. During sampling we assure that $s_{i-1} < s_i^* < s_{i+1}$. In this work, considering physical robot and task dimensions, we use radius 0.05 and $m = 15$. Then, we replace b_i with $b_{i,new}$ by minimising a cost $l(b_i)$:

$$\begin{aligned} b_{i,new} &= \underset{b_i^* \cup b_i}{\arg\min}\, l(b_i), \text{where} \\ l(b_i) &= \frac{\delta \mathbf{x}_{b,i+1} + \delta \mathbf{x}_{b,i}}{s_{i+1} - s_{i-1}} + \left| \frac{\delta \mathbf{x}_{b,i+1}}{\delta s_{i+1}} - \frac{\delta \mathbf{x}_{b,i}}{\delta s_i} \right| \end{aligned} \tag{5}$$

where $\delta \mathbf{x}_{b,i} = dist(\mathbf{x}_{b,i}, \mathbf{x}_{b,i-1})$ is computed using *RRT**'s distance function. The minimised function $l(b_i)$ is similar to how dynamic constraints are imposed in [47]. As print speed is constant, the first term of l is related to the robot-base velocity over the whole segment s_{i-1} to s_{i+1}. Similarly, the second term of l is related to acceleration. The new $b_{i,new}$ leads to lower and more constant velocity throughout the segment. This relaxation step is carried out 10 times.

B. SLQ-MPC Base Trajectory Reference

We use the following procedure to add the robot-base path b_i to the SLQ-MPC reference trajectory $\hat{\mathbf{x}}$ First, $\mathbf{x}_{b,i} \in b_i$ elements are transformed to SE(3) and stored as positions and quaternions assuming that the robot base motion is constrained to the xy plane. Using the process variable s_i, $\mathbf{x}_{b,i}$ is interpolated and time-stamped to match the print trajectory $\hat{\mathbf{p}}$. Finally, $\hat{\mathbf{x}}$ is extended to include both the print task and the robot-base reference $\hat{\mathbf{x}}(t) = (\hat{\mathbf{p}}(t), \hat{\mathbf{x}}_b(t))$.

We explored three ways to include $\hat{\mathbf{x}}_b(t)$ in the MPC controller cost function: quadratic tracking functions, time-varying soft constraints, and a mixture of the two.

a) Quadratic tracking functions: The base tracking cost C_b can be computed the same way as the end-effector tracking costs C_{ee} (Eq. 3), but only penalising the x, y and *yaw* elements of the robot-base state error:

$$C_b(t, \mathbf{x}, \hat{\mathbf{x}}) = [\mathbf{e}_{xy}, \mathbf{e}_{yaw}]^\intercal \mathbf{Q}_b [\mathbf{e}_{xy}, \mathbf{e}_{yaw}] \tag{6}$$

The shortcoming of this approach is that C_b directly competes with C_{ee}. If the weights Q_b are of comparable size to the C_{ee} weights Q_{ee}, the SLQ-MPC will effectively only solve the inverse kinematics problem between the print and robot-base trajectories. Also, this could mean compromising the end-effector error over robot-base path error. On the other hand, if Q_b is small, the controller could leverage robot-base for increased agility, but it could lead to a steady state error, while following the robot-base trajectory. As in this work, collision with the print task is taken into account only by the TCPP-derived robot-base path, this could mean driving over the printed structure.

b) Time-varying soft constraints: The base reference trajectory can also be modelled as a soft constraint. To do this, we first derive feasible regions around each b_i by further leveraging the functionality of TCPP. For each b_i, we incrementally perturb the base pose $\mathbf{x}_{b,i} = (x_i, y_i, \theta_i)$ along positive and negative SE(2) axes. We store the largest total increment along each axis for which the perturbed $\mathbf{x}_{b,i}$ can still be validated using the TCPP *isValid* routine. That is, if robot-base is not in collision and print path at s_i is reachable. For example:

$$\begin{aligned} r_{x,i} = \max_{\mathbb{N}}(N) \cdot \varepsilon, \quad \text{such that,} \\ isValid((x \pm n\varepsilon, y, \theta, s_i)) \forall n \leq N \end{aligned} \tag{7}$$

where the increments $\varepsilon = 0.03$ are determined experimentally. These maximum deviations $\mathbf{r}_{xy\theta,i} = (r_{x,i}, r_{y,i}, r_{\theta,i})$ effectively define an ellipsoid around b_i in SE(2) of feasible robot poses. $\mathbf{r}_{xy\theta}$ is interpolated and time-stamped along with b_i and added to the reference trajectory $\hat{\mathbf{x}} = (\hat{\mathbf{p}}, \hat{\mathbf{x}}_b, \mathbf{r}_{xy\theta})$. In turn, the violation of leaving this region $\mathbf{z}$ is then:

$$\mathbf{z} = \sigma(\mathbf{r}_{xy\theta}(t) - |\mathbf{x}_b(t) - \hat{\mathbf{x}}_b(t)|). \tag{8}$$

Leaving this region is penalised using a relaxed barrier function alongside other constraints as in Eq. 4. In this way, a robot-base reference pose is provided as a moving region, which still carries assurances from TCPP, but does not force the controller to compromise between competing tracking costs. σ is a tuning parameter. We use $\sigma = 25$ (determined experimentally).

c) Hybrid constraint: Quadratic cost draws the robot-base to nominal pose, whereas soft constraints allow unrestricted movement within a feasible region. These are complementary behaviours. Therefore, we introduce the hybrid method which sets the base tracking cost to be the sum of the quadratic and soft constraint terms. This creates bounded flexibility by only gently drawing the robot towards the TCPP-derived path, but penalising leaving the feasible region.

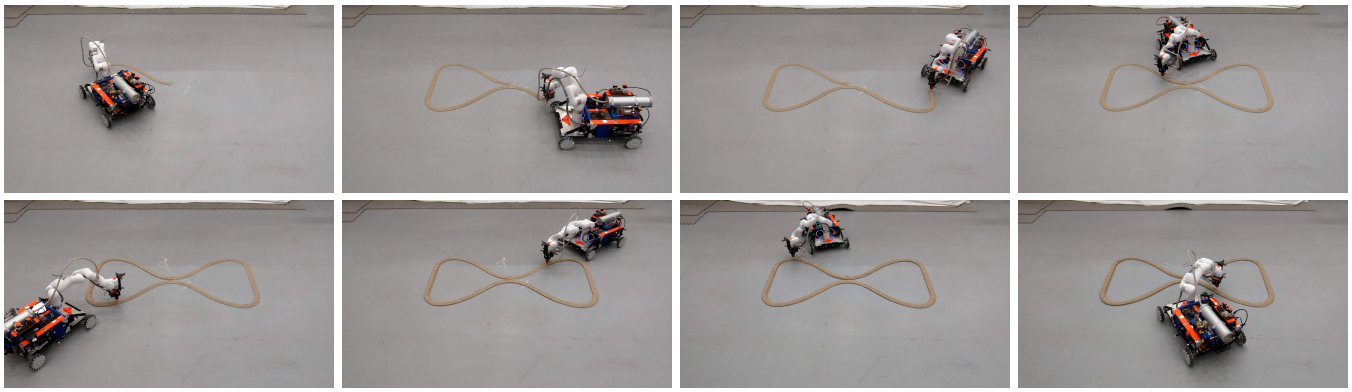


Fig. 4: Armstone robot printing the first 3 layers of the *bowtie* task. Note, the point of view of these images is facing along the negative y-axis when comparing to Fig. 6. Full video of experiments is available at `youtu.be/uBzdOuvIp9c`

V. Tuning and Hardware Error Sources

The SLQ-MPC end-effector tracking and system input weights are set as follows. The translational elements of $\mathbf{Q}_{ee}$ (see Eq. 3) are set to 250, while roll and pitch elements are set to 100. As the end-effector orientation about the extruder nozzle does not matter, the yaw weight is 0. Furthermore, the input costs $\mathbf{R}$ are set to 0.1 for all arm joint velocities, while base input weights are 10. The task-orientation $\hat{\mathbf{p}}_q$ remains constant pointing the end-effector along negative Z axis.

In tuning the system, we found two hardware-related sources of end-effector error. As the Armstone manipulator stretches away from the robot-base, mechanical components, e.g., aluminium extrusion frame or front wheel axles, experience increased load. In turn, this causes a slight drop in end-effector position. Using the OptiTrack we estimated the end-effector is about 4 mm lower at full arm stretch than when close to robot-base. Moreover, at the time of writing, the xArm manipulator firmware struggles to render smooth motion at the low joint velocities we use to perform printing. This effect is most pronounced and visible as vertical oscillations. To estimate the magnitude these oscillations, we performed straight line motions along x, y and z axes in the end-effector frame while the base is stationary. From the tracking data we found the mean and maximum L_2 norms of errors are 0.6 mm and 1.9 mm. We do not account for these errors and they are present in the results.

VI. Evaluation

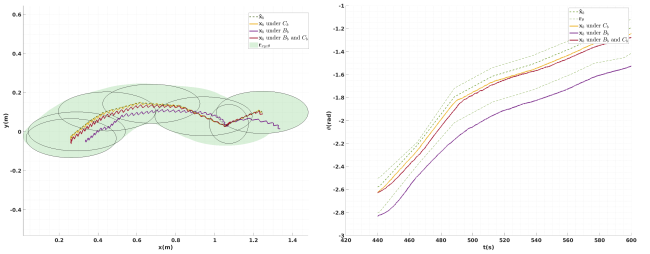
To evaluate the effectiveness of our approach we carried out two sets of large-scale printing experiments. In the first set, we empirically evaluate the three different ways that the robot-base trajectory can be integrated into SLQ-MPC. The second set of experiments explored the effects of using on-board sensors (odometry and SLAM) to localise the platform.

A. Long-Term Printing Experiments

In Sec. IV we introduced three ways to integrate the robot-base trajectory into SLQ-MPC: the quadratic cost C_b, the time-varying constraint B_b, and a hybrid approach which summed both together. In this experiment, we compared these different approaches for the large-scale print task shown in Fig. 4. The task is a bowtie-shaped task trajectory. It is formed of 5 layers of 6 cm wide cross-hatching fill patterns following a $2\,\text{m} \times 1\,\text{m}$ bowtie shape. The shape was chosen as it has inside and outside corners as well as straight segments. Each layer is 51.93 m long and is sampled at 2.5 mm intervals along the path integral. Therefore, the total path is approximately 260 m. The printing speed is $3\,\text{cm}\,\text{s}^{-1}$. Each layer takes 28 min to print. To isolate controller behaviour from localisation errors an OptiTrack system was used to provide the base pose estimate.

When only C_b is used, the weights $\mathbf{Q}_b$ were set to 20 in order to assure close following of the base-path reference. When C_b is used together with B_b, values of $\mathbf{Q}_b$ are reduced to 2. Tuning of B_b was kept constant as in Sec. IV. The end-effector tracking costs C_{ee} are active at all times and are weighted as in Sec. III-B. Physical printing was only carried out for the hybrid costs formulation and is seen in Fig. 4. The reference trajectory $\hat{\mathbf{x}}$ and is sent to the SLQ-MPC controller one layer at a time.

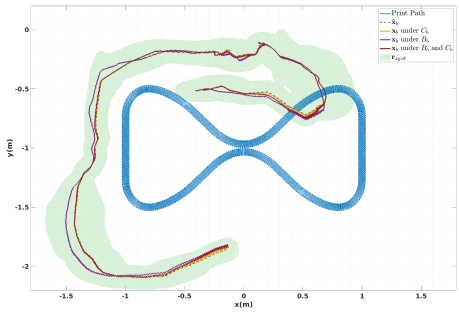
1) Base Reference Tracking: Fig. 6 shows the first three layers of the task solved by the TCPP algorithm. When Layer 1 is printed, the space gradually becomes obstructed by printing. When printing Layers 2 and 3, the print pattern is effectively a static obstacle. The task differs slightly between even and odd layers as the cross-hatching print pattern tilts in alternating directions to aid area coverage. This is commonly used technique in additive manufacturing. Fig. 5 shows a close up segment of layer 3 and illustrates the behaviour of the different cost function formulations. In Fig. 5a the robot-base pose falls behind the reference trajectory when only B_b



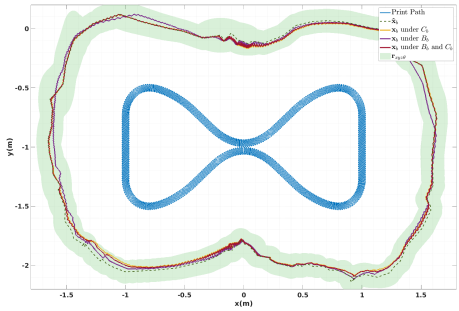
(a) Robot-base position (b) Robot-base orientation

Fig. 5: A segment of robot-base paths for layer 3 of the bowtie task.

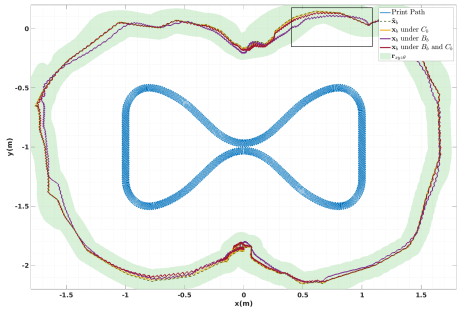
is used. The small constraint violation seen in Fig. 5b is pushing the robot-base as the boundary moves. The nature of the task requires robot to move around it. This results in an often small orientation constraint violations. The addition of C_b leads to the base path following the reference trajectory closer, but still allows deviations within the moving $\mathbf{r}_{xy\theta}$ region.



(a) Layer 1



(b) Layer 2



(c) Layer 3. Highlighted region is the source of Fig. 5

Fig. 6: Armstone-base motion when printing the bowtie task. Base path reference $\hat{\mathbf{x}}_\mathbf{b}$ and base paths as followed by the controller. Light green region illustrates the barrier function tolerance region $\mathbf{r}_{xy\theta}$

Furthermore, Fig. 6 shows the $\mathbf{r}_{xy\theta}$ regions shown in light green. The regions are more generous during the first layer as the robot prints behind itself and has more obstacle free space to move. The B_b-only exploits this by deviating further from reference trajectory. However, as the reference trajectory makes a turn and $\mathbf{r}_{xy\theta}$ follows, B_b-only paths make a sharp adjustment. This is seen in upper left region in Fig. 6a. The combined B_b and C_b paths deviate further from the reference trajectory than a C_b-only, but rarely come close to B_b constraint boundary. This indicates that $\mathbf{Q}_b$ could be reduced further. However, as presented next, we did not find it to be of large significance to the end-effector error.

2) End-Effector Trajectory Tracking: The end-effector error was computed using the OptiTrack ground truth data. Fig. 7 shows a close up of the traced print path when both cost terms are active. The distance between step function-like infill lines seen in Fig. 7 is 1 cm. The end-effector is seen occasionally missing the line and overlapping a previous one.

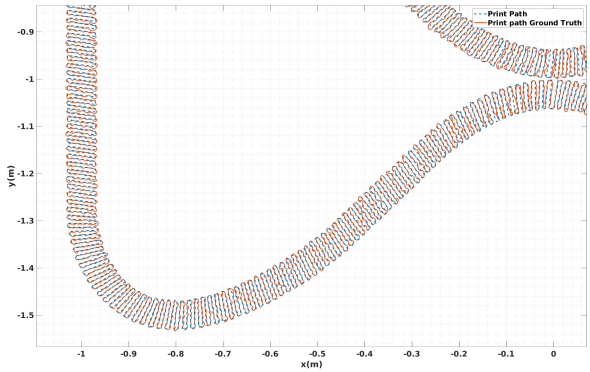


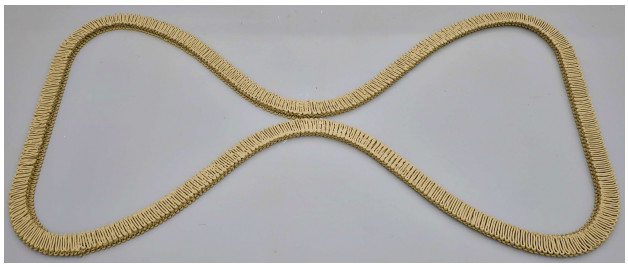
Fig. 7: Close up of end-effector path when C_b and B_b are active

Table I presents layer-by-layer average translational end-effector error ($\left\|\hat{\mathbf{p}}_{xyz} - \tilde{\mathbf{p}}_{xyz}\right\|_2$) for all experiments where $\tilde{\mathbf{p}}$ is the independently measured end-effector ground truth. It also shows an Iterative Closest Point (ICP) [48] derived error. This is also computed as an average L_2 error, but after firstly point matching and aligning $\hat{\mathbf{p}}_{xyz}$ to $\tilde{\mathbf{p}}_{xyz}$. The data suggests the hybrid cost-constraint approach has slightly lower end-effector average error than quadratic costs only. When printing with B_b constraint only, the average and maximum error varies between layers likely due to interactions with B_b boundary. When printing layer 4 using combined costs, the end-effector made a small aggressive motion penetrating the print. This is seen in the highlighted 36 mm maximum error. As this was a single occurrence and we were able to continue printing. The ICP error shows less significant impact between the robot-base path tracking costs and the resulting end-effector path tracking error. The discrepancy between the two error calculations presented indicates either a presence of a steady-state trajectory tracking error or experimental setup limitations arising from poor measurement synchronisation. The average roll and pitch error throughout all experiments was about 0.03 rad.

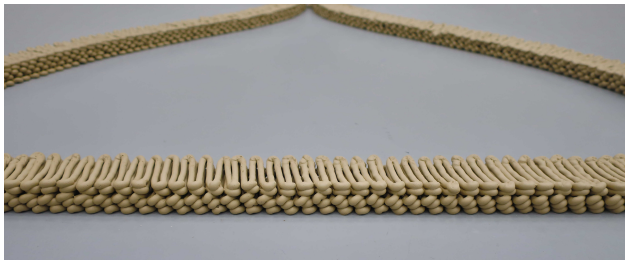
	ICP (mm)	mean (std) (mm)	max (mm)
L1, C_b	2.37	8.57 (1.96)	16.57
L2, C_b	2.73	8.40 (1.76)	13.88
L3, C_b	2.68	8.35 (2.07)	16.34
L1, B_b	2.39	6.66 (1.85)	16.39
L2, B_b	2.47	8.63 (1.83)	20.23
L3, B_b	2.61	2.78 (1.20)	10.57
L1, C_b,B_b	2.09	6.75 (1.58)	19.04
L2, C_b,B_b	2.6	6.87 (1.64)	22.37
L3, C_b,B_b	2.46	2.99 (1.32)	12.97
L4, C_b,B_b	2.47	6.85 (2.72)	**36.69**
L5, C_b,B_b	2.5	6.87 (2.65)	20.12

TABLE I: End-effector translational error

The final result of physically printing of the bowtie task is presented in Fig. 8. In total about 10 L of clay was used and the Armstone material cylinder was refilled after every



(a) Top view



(b) Close up side view

Fig. 8: Final result of printing the long bowtie print task. The task consists of five 51.93 m long layers spanning 2 m × 1 m in scale.

layer. As fine tuning material viscosity, extrusion rate and layer height is out of scope for this work, these parameters were adjusted experimentally. The layer height was set to 6 mm and print path was elevated by 1 cm above the layer to allow for inaccuracies in nozzle motion.

B. Printing with On-Board Localisation

High accuracy nozzle localisation is crucial to 3D printing, but highly challenging to achieve with a mobile manipulator system. To examine how printing fidelity is affected when using on-board SLAM or wheel odometry we carried out a second set of experiments. For this we used a 36.8 m long, 2.5 m × 0.3 m scale, two layer sine wave task composed of the same infill pattern as the bowtie task. Printing was carried out using the hybrid cost formulation.

The SLAM implementation from Sec. III-A was used to provide wheel odometry correction in the form of *map* to *odometry* frame offsets. This correction could vary several centimetres from update to update and could cause discontinuities in SLQ-MPC state feedback. Therefore we used a 5 s moving average filter to smooth it.

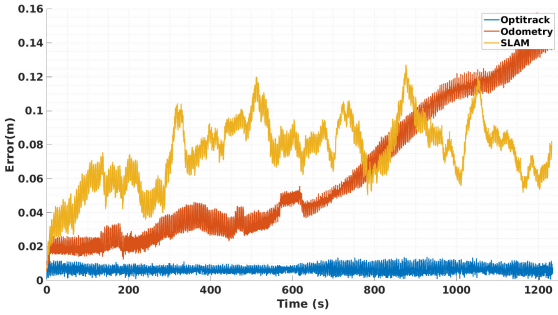


Fig. 9: End-effector errors for different localisation methods.

Fig. 9 shows that the end-effector error remains bounded within 0.1 m when printing using SLAM. This is expected

(a) Localisation using external tracking

(b) Localisation using wheel odometry

(c) Localisation using SLaM

Fig. 10: Comparison of printing performance using different localisation methods. Printed task is 2.5 m × 0.3 m scale. Task starts on the left, proceeds right and comes back.

given the map resolution and Armstone's LiDAR error of 3 cm. The print in Fig. 10 confirms this. However, the print also shows that using SLAM while printing can cause problems. As the map is updated with more observations, the localisation errors change. As a result, the different layers do not precisely overlap one another and the infill pattern shows higher deformation.

VII. Conclusions and Future Work

In this paper we have demonstrated Autonomous Mobile 3D Printing of large-scale trajectories. Given only the print task and robot kinematic model, the TCPP algorithm was used to derive a feasible robot base path and in turn inform a SLQ-MPC motion controller. To the best of authors knowledge, the over 250 m long bowtie print experiment carried out in Sec. VI is the largest and highest autonomy print done with a moving mobile manipulator to date.

Notably, the nominal end-effector accuracy of 6 mm - 8 mm was achieved via external tracking. Reliance on external tracking can be avoided by adopting fiducial marker [19] or end-effector mounted laser range finding [49].

In future work, we will also explore simultaneous printing and mapping to localise the end-effector with respect to the print task [50]. Furthermore, granted end-effector localisation, the MPC motion controller and the TCPP path planner can both already support printing of more complex, non-planar structures. This, together with scaling up the Armstone robot for deployment outside lab conditions has the potential to drastically increase system applicability.

References

[1] N. Labonnote, A. Rønnquist, B. Manum, *et al.*, "Additive Construction: State-of-the-Art, Challenges and Opportunities," *Automation in Construction*, vol. 72, pp. 347–366, dec 2016.

[2] T. Bock, "The future of construction automation: Technological disruption and the upcoming ubiquity of robotics," *Automation in Construction*, vol. 59, pp. 113–121, nov 2015.

[3] Arup, "Sagrada Familia," 2019. [Online]. Available: https://www.arup.com/projects/sagrada-familia

[4] M. Pauli, "Prefabrication and modular construction – from fragmented to integrated value chains," Tech. Rep., 2009.

[5] Arup, "Can drones transform surveying and modelling?" 2015. [Online]. Available: https://research.arup.com/projects/can-drones-transform-surveying-and-modelling/

[6] K. Booth, "Arup uses UAVs to undertake survey of Lloyd's Building external facade," 2021. [Online]. Available: https://www.bdcmagazine.com/2021/03/arup-uses-uavs-to-undertake-survey-of-lloyds-building-external-facade/

[7] Y. W. D. Tay, B. Panda, S. C. Paul, *et al.*, "3D printing trends in building and construction industry: a review," *Virtual and Physical Prototyping*, vol. 12, no. 3, pp. 261–276, jul 2017.

[8] F. Bos, R. Wolfs, Z. Ahmed, *et al.*, "Additive manufacturing of concrete in construction: potentials and challenges of 3D concrete printing," *Virtual and Physical Prototyping*, vol. 11, no. 3, pp. 209–225, jul 2016.

[9] H. Zhong and M. Zhang, "3D printing geopolymers: A review," *Cement and Concrete Composites*, no. February, p. 104455, feb 2022.

[10] S. Hou, Z. Duan, J. Xiao, *et al.*, "A Review of 3D Printed Concrete: Performance Requirements, Testing Measurements and Mix Design," 2021.

[11] COBOD, "Modular 3D Construction Printers," 2018. [Online]. Available: https://cobod.com/

[12] Apis Cor, "Apis Cor - we print buildings," p. 26, 2017. [Online]. Available: https://www.apis-cor.com/http://apis-cor.com/en/3d-printer

[13] Cybe, "We redefine construction," 2020. [Online]. Available: https://cybe.eu/

[14] HSE, "Construction statistics in Great Britain in the Construction sector," pp. 1–20, 2020. [Online]. Available: http://www.hse.gov.uk/statistics/industry/construction.pdf

[15] K. Dörfler, G. Dielemans, L. Lachmayer, *et al.*, "Additive Manufacturing using mobile robots: Opportunities and challenges for building construction," *Cement and Concrete Research*, vol. 158, aug 2022.

[16] X. Zhang, M. Li, J. H. Lim, *et al.*, "Large-scale 3D printing by a team of mobile robots," *Automation in Construction*, vol. 95, pp. 98–106, nov 2018.

[17] J. Sustarevas, D. Butters, M. Hammid, *et al.*, "MAP - A Mobile Agile Printer Robot for on-site Construction," in *International Conference on Intelligent Robots and Systems*, 2018, pp. 2441–2448.

[18] J. Sustarevas, D. Kanoulas, and S. Julier, "Task-Consistent Path Planning for Mobile 3D Printing," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 2143–2150.

[19] M. E. Tiryaki, X. Zhang, and Q. C. Pham, "Printing-while-moving: A new paradigm for large-scale robotic 3D Printing," in *International Conference on Intelligent Robots and Systems*, 2019, pp. 2286–2291.

[20] G. Dielemans, L. Lachmayer, T. Recker, *et al.*, "Mobile Additive Manufacturing: A Case Study of Clay Formwork for Bespoke in Situ Concrete Construction," in *Third RILEM International Conference on Concrete and Digital Fabrication*. Springer, Cham, 2022, pp. 15–21.

[21] S. J. Keating, J. C. Leland, L. Cai, *et al.*, "Toward site-specific and self-sufficient robotic fabrication on architectural scales," *Science Robotics*, vol. 2, no. 5, p. eaam8986, apr 2017.

[22] J. Sustarevas, K. X. Benjamin Tan, D. Gerber, *et al.*, "YouWasps: Towards Autonomous Multi-Robot Mobile Deposition for Construction," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 2320–2327.

[23] S. Karaman, M. R. Walter, A. Perez, *et al.*, "Anytime motion planning using the RRT," *Proceedings - IEEE International Conference on Robotics and Automation*, no. June 2011, pp. 1478–1483, 2011.

[24] S. M. LaValle, *Planning Algorithms*. Cambridge: Cambridge University Press, 2006, vol. 9780521862.

[25] V. S. Raghavan *et al.*, "Variable Configuration Planner for Legged-Rolling Obstacle Negotiation Locomotion: Application on the CENTAURO Robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 4738–4745.

[26] ——, "Agile Legged-Wheeled Reconfigurable Navigation Planner Applied on the CENTAURO Robot," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 1424–1430.

[27] ——, "Reconfigurable and Agile Legged-Wheeled Robot Navigation in Cluttered Environments With Movable Obstacles," *IEEE Access*, vol. 10, pp. 2429–2445, 2022.

[28] G. Oriolo, M. Ottavi, and M. Vendittelli, "Probabilistic motion planning for redundant robots along given end-effector paths," in *IEEE/RSJ International Conference on Intelligent Robots and System*, 2002, pp. 1657–1662.

[29] G. Oriolo and C. Mongillo, "Motion planning for mobile manipulators along given end-effector paths," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2005, no. April, pp. 2154–2160, 2005.

[30] K. Nagatani, T. Hirayama, A. Gofuku, *et al.*, "Motion planning for mobile manipulator with keeping manipulability," *IEEE International Conference on Intelligent Robots and Systems*, vol. 2, no. October, pp. 1663–1668, 2002.

[31] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Robot placement based on reachability inversion," in *2013 IEEE International Conference on Robotics and Automation*, no. 2. IEEE, may 2013, pp. 1970–1975.

[32] F. Zacharias, W. Sepp, C. Borst, *et al.*, "Using a Model of the Reachable Workspace to Position Mobile Manipulators for 3-D Trajectories," in *9th IEEE-RAS Int. Conf. on Humanoid Robots*, 2009, pp. 55–61.

[33] F. Zacharias, C. Borst, and G. Hirzinger, "Capturing Robot Workspace Structure: Representing Robot Capabilities," in *IEEE International Conference on Intelligent Robots and Systems*, 2007, pp. 3229–3236.

[34] F. Gramazio and M. Kohler, "In - situ fabrication mobile robotic units on construction sites," ETH Zurich, Tech. Rep., 2014.

[35] E. Jelavic, Y. Berdou, D. Jud, *et al.*, "Terrain-Adaptive Planning and Control of Complex Motions for Walking Excavators," *IEEE/RSJ Intenational Conference on Intelligent Robots and Systems*, 2020.

[36] F. Farshidian, M. Neunert, A. W. Winkler, *et al.*, "An Efficient Optimal Planning and Control Framework for Quadrupedal Locomotion," in *IEEE Int. Conference on Robotics and Automation*, 2017, pp. 93–100.

[37] A. Laurenzi, D. Kanoulas, E. Mingo Hoffman, *et al.*, "Whole-Body Stabilization for Visual-Based Box Lifting with the COMAN+ Robot," in *3rd IEEE Int. Conf. on Robotic Computing*, mar 2019, pp. 445–446.

[38] J. Pankert and M. Hutter, "Perceptive model predictive control for continuous mobile manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6177–6184, oct 2020.

[39] F. Farshidian, "Optimal Control for Switched Systems." [Online]. Available: https://leggedrobotics.github.io/ocs2/

[40] M. Frigerio, J. Buchli, D. G. Caldwell, *et al.*, "RobCoGen : a code generator for efficient kinematics and dynamics of articulated robots , based on Domain Specific Languages," *Journal of Software Engineering for Robotics*, vol. 7, no. July, pp. 36–54, 2016.

[41] B. Siciliano, L. Sciavicco, L. Villani, *et al.*, *Robotics*, ser. Advanced Textbooks in Control and Signal Processing. London: Springer London, 2009.

[42] C. Feller and C. Ebenbauer, "Relaxed Logarithmic Barrier Function Based Model Predictive Control of Linear Systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 3, pp. 1223–1238, mar 2017.

[43] Iralabdisco, "ira_laser_tools." [Online]. Available: https://github.com/iralabdisco/ira_laser_tools

[44] S. Macenski and I. Jambrecic, "SLAM Toolbox: SLAM for the dynamic world," *Journal of Open Source Software*, vol. 6, no. 61, p. 2783, may 2021.

[45] S. Chitta, E. Marder-Eppstein, W. Meeussen, *et al.*, "ros_control: A generic and simple control framework for ROS," *The Journal of Open Source Software*, vol. 2, no. 20, p. 456, 2017.

[46] M. Kanehara, S. Kagami, J. J. Kuffner, *et al.*, "Path shortening and smoothing of grid-based path planning with consideration of obstacles," in *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, 2007, pp. 991–996.

[47] C. Rösmann, W. Feiten, T. Wösch, *et al.*, "Trajectory modification considering dynamic constraints of autonomous robots," in *7th German Conference on Robotics, ROBOTIK 2012*, 2012, pp. 74–79.

[48] P. J. Besl and N. D. McKay, "A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[49] A. Gawel, R. Siegwart, M. Hutter, *et al.*, "A Fully-Integrated Sensing and Control System for High-Accuracy Mobile Robotic Building Construction," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, nov 2019, pp. 2300–2307.

[50] J. Li, P. L. Aubin-Fournier, and K. Skonieczny, "SLAAM: Simultaneous Localization and Additive Manufacturing," *IEEE Transactions on Robotics*, pp. 1–16, 2020.