PLASTR: Planning for Autonomous Sampling-Based Trowelling

Mads A. Kuhlmann-Jørgensen, Johannes Pankert, Lukasz L. Pietrasik, and Marco Hutter, *Member, IEEE*

***Abstract*—Plaster is commonly used in the construction industry to finish walls and ceilings, but the application is labor-intensive and physically strenuous, which motivates the need for automation. We present PLASTR, a receding horizon optimization-based planning algorithm for robotic plaster trowelling. It samples trowelling sequence rollouts from a new plaster simulator and weights them according to the flatness of the finished wall. The proposed simulator approximates the real-world plaster-trowel interaction adequately while allowing execution orders of magnitude faster than real-time. We evaluate PLASTR in simulation and on a real-world test setup and compare it to two handcrafted heuristic baseline algorithms. PLASTR performs equal to or better than the best heuristic in terms of material coverage for both simulated and real-world experiments while being 50% more efficient in terms of trowelled distance.**

***Index Terms*—Robotics and automation in construction, motion and path planning, optimization and optimal control.**

I. Introduction

DESPITE the construction industry being slow to adopt innovation, the field of construction robots has seen significant growth in recent years due to the potential economic, operational, and environmental benefits of bringing automation to the building site [1]. Plasterwork is a very common task in construction due to various desirable properties of plaster, including fire protection, heat and humidity regulation, and aesthetics [2], [3]. The common process for plastering is to first coarsely apply plaster onto the surface and afterwards iteratively spreading the material using a plastering trowel until the desired surface is obtained. This work is labor intensive and physically strenuous, motivating automation in this area to bring benefits in economic terms and regarding worker health and safety. However, craftsmen typically rely on years of experience and a heavy amount of intuition based on visual and interaction feedback, making it challenging to transfer this task to a robotic platform. In this paper, we present the findings of our work towards automating trowel-based plastering, with the long-term goal of producing surfaces of human worker-level quality. In this foundational work, we focus on the task of even distributing pre-applied plaster across a wall surface.

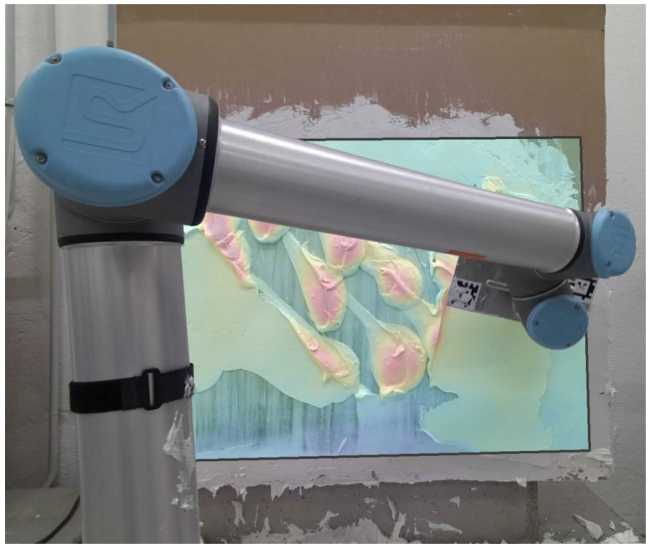


Fig. 1. A robotic manipulator is smoothening a surface with troweling actions. The work area is marked, and the measured elevation is visualized with the overlaid color map.

A. Related Work

The subject of robotic plastering has been studied for more than 25 years, with early works examining manipulator design, mapping, and navigation for robotic plaster spraying [4], [5]. Another early work holistically analyzes different mechanical components required when designing a plastering robot, e.g., the manipulator end-effector [2]. Much of the literature on robotic plastering investigates mechanical design and low-level control of the manipulator and end-effector [6], [7], [8]. For the latter, several works propose leveraging a laser or a similar depth sensor mounted on or near the trowel to ensure correct positioning relative to the surface of interest [9], [10]. [11] additionally include a torque sensor in a 3DoF end-effector design, allowing for impedance control to be utilized in keeping sufficient surface contact.

The area of trajectory generation and path planning for robotic plastering is less covered in existing literature, especially for

Manuscript received 9 March 2023; accepted 21 June 2023. Date of publication 3 July 2023; date of current version 11 July 2023. This letter was recommended for publication by Associate Editor L. Biagiotti and Editor L. Pallottino upon evaluation of the reviewers' comments. This work was supported by the Swiss National Science Foundation through the National Centre of Competence in Digital Fabrication (NCCR dfab) and Innosuisse under Grant 51928.1 *(Corresponding author: Mads A. Kuhlmann-Jørgensen.)*

The authors are with the Robotic Systems Lab., ETH Zurich, 8092 Zurich, Switzerland (e-mail: makuhlmann@ethz.ch; pankertj@ethz.ch; lukaszpi@ethz.ch; mahutter@ethz.ch).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2023.3291894, provided by the authors.

Digital Object Identifier 10.1109/LRA.2023.3291894

trowel-based plastering. Methods have been proposed for automatically adapting trajectories to the current state of the work surface [12], [13] and for an operator to interactively define trajectories in an augmented reality setting [14]. However, both of these works only consider plaster spraying. For trowelling, the primary investigation has been in training a convolutional neural network to predict trowelling directions based on images of the surface [15], [16]. Since little prior work has been conducted on path planning for robotic plaster trowelling, this is an unexplored field open to investigation.

While the literature on action planning, specifically for the manipulation of plaster, is sparse, robotically shaping other complex materials, where the outcome of a given action is inertly hard to predict, has been investigated. Molds have been fabricated by iteratively stamping a clay block into the desired shape based on a heuristic [17]. Another work uses a roller to shape clay-like material to fit a target 2D contour [18]. Material is moved outwards from the point with the largest material excess. We compare our approach to this strategy in the evaluation section. Similar heuristics have been applied to plan autonomous excavation [19]. The dig- and dump positions for shaping soil are computed from metrics such as target deviation and current bucket position. These works all employ a feedback-driven approach: Actions are planned and executed iteratively based on the current material state, aiming to converge to the target shape.

Data-driven approaches for selecting actions have become popular in recent years. Imitation learning from human demonstrations has been used to teach a robot to shape kinetic sand [20]. Manipulation of granular media can be learned with deep reinforcement learning from real-world data collected on a physical experimentation setup [21]. Due to the large data requirement of learning-based approaches, many researchers use simulators for training. Some approaches explicitly learn the action policy [22], [23], while others learn the physics model and define the action policy based on optimization [24], [25]. If the learned physics model is differentiable, this can be exploited by gradient-based optimization [24]. Path Integral Policy Improvement (PI^2) can be used when explicit gradients are not available. It finds optimal actions based on value function gradients estimated from sampled action sequence roll-outs [26], [27]. We use PI^2 in this work to find optimal trowel paths based on simulator roll-outs.

Even though simulating viscous and granular material is challenging, commercial software is available for the task. A popular and versatile particle-based simulator for granular media and fluids is nVidia FleX [28]. It runs in real-time rendering a single frame in $10\,\text{ms}$. Other simulators are more accurate but come at a higher computational cost [29]. Viscoelastic materials, in particular, are expensive and time-consuming to simulate. Recent methods do, however, allow for real-time execution at $0.03\,\text{s}$ per frame with 15 k particles [30]. While this might be sufficient to facilitate offline learning, we intend to use simulation online for stochastic optimization. This requires orders of magnitude faster execution which is why we propose a new trowelling simulator powered by lightweight interaction heuristics.

B. Contributions

We provide the following contributions to the literature:

1) Development of a fast plaster trowelling simulator for optimization-based planning.
2) Proposing PLASTR, an optimization-based planning algorithm for trowelling.
3) Evaluation of the simulator and the PLASTR strategy in extensive robotic plastering experiments.

II. Trowelling Simulator

We introduce a plaster trowelling simulator for action planning based on stochastic, sampling-based optimization. To achieve reasonable planning times, the simulator's speed should be orders of magnitude faster than real-time. Therefore, we focus on speed over accuracy by developing a volume-based simulator specifically for trowelling actions. While a sim-to-real gap is expected, we foresee that this will be addressed by the closed-loop performance of our planner as we want to replan continuously.

The simulator is based on the movement of a trowel through a 2.5 d grid-based elevation map environment, where each cell represents the elevation as a continous value. The Grid Map library is utilized in the software backend [31]. For each simulation time step, the effect on the swept area of the plastering tool between two nearby locations is computed. The full algorithm is described in Algorithm 1. During the each step, the trowel can *accumulate* or *deposit* material:

A. Accumulation

The material in the swept region and above the tool elevation should be scraped off and accumulated. The first is achieved by setting the elevation in the relevant grid cells equal to the trowel height, and the second is realized by adding the removed material to a buffer. The buffer is divided into several bins as illustrated in Fig. 2 to emulate the distribution of the accumulated material across the width of the tool. The buffer cannot hold infinite amounts of material. The accumulation is bounded by the available space between the trowel and the wall as shown in Fig. 2. We call the space accumulation threshold τ_{acc} and compute it as follows:

$$\tau_{acc} = w \cdot \frac{l^2 \cdot \sin(\alpha) \cdot \cos(\alpha)}{2} \tag{1}$$

w and l are the width and length of the tool, α is the pitch angle. Note that the threshold is adaptive and depends on α. Since only a very small region is considered at a time, we assume α, and thus τ_{acc}, to be constant. The added material can exceed τ_{acc} (Algorithm 1, line 6), however, due to the volume of a single cell, this is considered negligible.

B. Deposition

Grid cells below the trowel height are filled with material accumulated on the tool. If the relevant bin holds enough accumulated material, the elevation of the grid cell is set to the

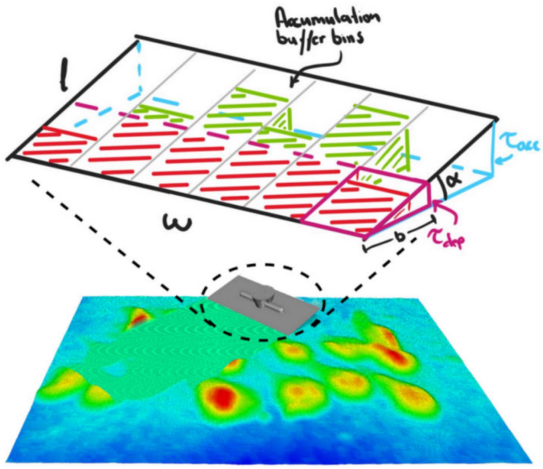


Fig. 2. The simulation model subdivides the plaster accumulation buffer on the trowel into bins. A maximum volume of τ_{acc} can be stored. During the material deposition phase, we assume that an adaptive volume τ_{dep} of plaster remains on the trowel, and excess material (green) can be deposited onto the surface.

Algorithm 1: Conceptual overview of one step of the simulator. `tool[c]` refers to the elevation of the trowel edge above the grid coordinates of cell `c`. `buffer` refers to all bins of the accumulation buffer. `bin[c]` indicates the specific bin which `c` projects into. `elevation[c]` refers to the material elevation at `c`.

Input:
– Two tool poses, p_1 and p_2
Initialize:
– Ω as the area covered by the tool between p_1 and p_2
– τ_{acc} and τ_{dep} as in (1) and (2) respectively

```
for all cells, c, in Ω do
    d ← elevation[c] − tool[c]
    if d > 0 then
        elevation[c] ← tool[c]
        if sum(buffer) < τ_acc then
            bin[c] ← bin[c] + d ▷ Accumulation
    else
        if bin[c] > τ_dep then
            elevation[c] ← tool[c] ▷ Deposition
            bin[c] ← bin[c] + d
buffer ← convolve(buffer)
```

trowel height, and the deposited volume is removed from the bin. Similar to accumulation, we propose an adaptive depositing threshold τ_{dep} that depends on the pitch of the tool:

$$\tau_{dep} = \frac{w}{n_{bins}} \cdot \frac{b^2 \cdot \tan(\alpha)}{2} \tag{2}$$

α and w follow the definitions from (1). n_{bins} is the number of accumulation bins. b is a design parameter that defines a distance from the trowel edge, parallel to the surface, to which the buffer should be filled before depositing. This heuristic leads to more accumulated material being required for depositing with a larger pitch angle which replicates the behavior observed in real experiments. To deposit material in a cell, the appropriate bin must hold at least the missing amount in the cell plus τ_{dep}. We observe that including the threshold and applying it per bin rather than for the whole trowel improves the realism of the simulation results.

A smoothing convolution operation is applied over the accumulation bins to simulate material diffusion between these. The lateral material spill is not simulated. Including this will improve the realism of the simulation and could be considered for future work. The kernel is a design parameter for which we choose an approximated Gaussian, $\omega = 0.25\,[1 \quad 2 \quad 1]$. Including the convolution, which is performed at each time step, qualitatively improved the simulation. Other kernels have not been examined.

TABLE I
WE PROVIDE SIMULATION ACCURACY METRICS FOR THE AFFECTED AREA OF SINGULAR SWEEPS ON A VERTICAL AND A HORIZONTAL SURFACE ALONG WITH A MEASUREMENT NOISE REFERENCE. *AREA WITHIN* $\pm 2\,\mathrm{mm}$ REFERS TO THE FACTION OF THE CONSIDERED AREA THAT FALLS INTO A TOLERANCE BAND THAT CAPTURES 99% OF THE NOISE

	$\mu \pm \sigma$ [mm]	RMSE [mm]	Area within $\pm 2\,\mathrm{mm}$
Vertical	0.08 ± 2.00	1.95	0.70
Horizontal	0.37 ± 1.42	1.38	0.84
Noise	-0.04 ± 0.95	0.78	0.99

C. Evaluation

We evaluate the developed simulator against real-world measurements from an Azure Kinect depth sensor. We define the grid resolution of the elevation map such that each cell covers $3 \times 3\,\mathrm{mm}$ and set the accumulation buffer bin width to be $1.75\,\mathrm{cm}$. The elevation map is initialized with the measured state of the real world before each trowelling stroke. The results are presented in Table I. The accuracy metrics are computed based on the difference between the surface predicted by the simulator and a real-world measurement after a single trowelling stroke. Fig. 3 shows examples of predictions and measurements. Only the area covered by each sweep is used in the computation, as the remaining area is expected to stay mostly unchanged. The included noise measure is computed based on the area not covered by each action. While in the worst case, the Root Mean Square Error (RMSE) deviation from the desired surface is 1.95 mm, 70% of the troweled area achieves the target height within a 2 mm tolerance. This shows that the proposed simulator is a reasonable approximation to the real world and we expect the deviations to be compensated for by the receding horizon nature of PLASTR. Execution time is a key feature of the simulator. Since this primarily depends on the length of the simulated sweep, we present the average time per trowelled meter. At $2.44\,\mathrm{ms/m}$, this is approximately 4000 times faster than the trowelling speed of $0.10\,\mathrm{m/s}$ used in our real-world experiments. It should be noted that the simulation speed can be further optimized through parallelization and utilization of GPU resources [32].

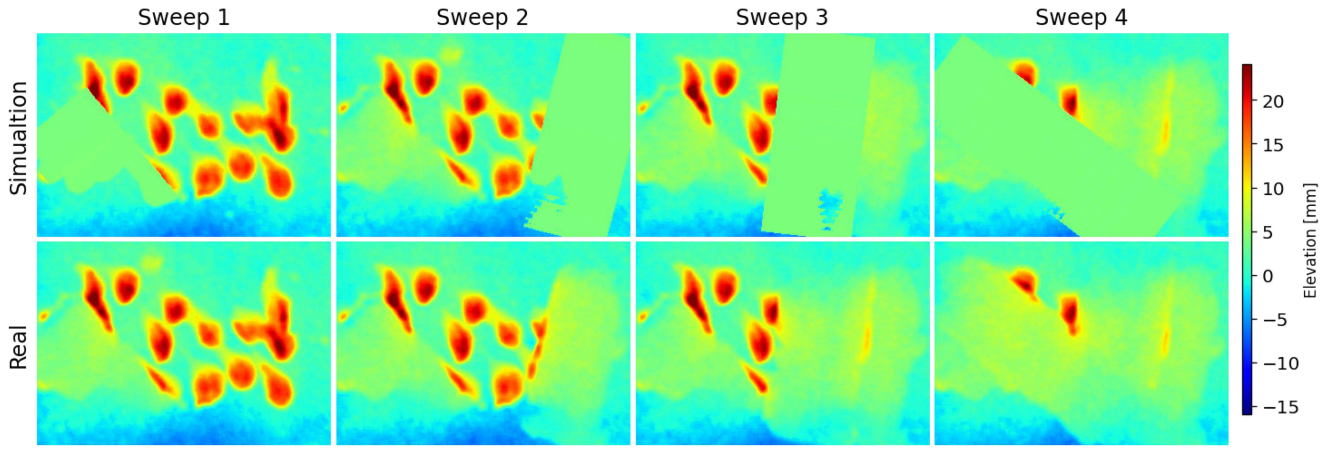


Fig. 3. Visualisation of the real measurements (bottom) and the one-step prediction based on the simulator (top). The color map indicates the elevation with the color gradient centered around the desired surface.

III. Planning Algorithms

A. State and Action Space

An elevation map reflecting the current state of the work area makes up the primary observation space utilized by all presented planners. This is updated from sensor measurements after each action. The optimization-based planner further requires an estimate of the accumulated material defined in Section II-A. The target surface is also represented as an elevation map. For the task of evenly spreading plaster, we consider only planar target surfaces. Their elevation is computed based on the volume of available plaster at the beginning of execution:

$$z_{target} = \frac{1}{m \cdot n} \cdot \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [G(i,j)] - \frac{\nu}{A} \tag{3}$$

G is the current elevation map, m and n are the rows and columns of G, A is the area of G, and ν is a design parameter to compensate for excess material. We define actions as linear trowel sweeps parameterized by a start and an endpoint, with the trailing edge of the trowel always perpendicular to the direction of motion. The start and end points are always placed in the target plane, thus leaving only four parameters $((x_{start}, y_{start}), (x_{goal}, y_{goal}))$. The pitch angle ($\alpha$ in Fig. 2) is kept constant for a single action. It is the smallest angle that still ensures that the upper edge of the trowel is free of collision with plaster. We favor smaller angles as experiments have shown these to yield better adhesion when depositing. This is also reflected in simulation by the adaptive threshold τ_{dep} which grows with the pitch angle.

B. Optimisation-Based Planning

With an optimization-based planning approach, we utilize the developed simulator to alleviate the need for hand-designed rules and heuristics. To circumvent the need for a differentiable simulator, we adopt a gradient-free path integral optimization scheme and base our implementation on the *STOMP* algorithm [27]. The algorithm starts from an initial sub-optimal and/or infeasible trajectory, which is iteratively updated until convergence based on noisy exploration samples. At each iteration, K noisy realizations of the current trajectory are sampled, and each time step of each sample is assigned a cost. The new trajectory is then formed by updating the current one based on a weighted average of the samples. The weights are inversely proportional to the costs.

Our method is outlined in Algorithm 2. It differs from *STOMP* in a few key areas:

Algorithm 2: We propose PLASTR, an optimization-based planning algorithm for plaster trowelling inspired by STOMP. Starting from an initial guess, the plan is updated based on weighted averages of noisy exploration samples. Following the notation introduced in the STOMP paper, we present the algorithm for one-dimensional actions.

Given: – Current state, s
– Horizon length, N
– Standard deviation for exploration noise, σ
Initialise action sequence
– If first run, initialise all N actions, $\theta_{1:N}$
– Else, shift previous θ and initialise only θ_N
Repeat until cost converges:
1) Create K action sequences: $\tilde{\theta}_k = \theta + \epsilon_k$, $\epsilon_k \sim \mathcal{N}(0, \boldsymbol{I} \cdot \sigma^2)$
2) **For** $k = 1 \dots K$:
a) Simulate $\tilde{\theta}_k$ starting from s to get ϕ_k and $q(\tilde{\theta}_{k,i})$
b) For each action find: $S(\tilde{\theta}_{k,i}) = \phi_k + \sum_{n=i}^{N} q(\tilde{\theta}_{k,n})$
c) Calculate $P(\tilde{\theta}_{k,i}) = \frac{e^{-\frac{1}{\lambda} S(\tilde{\theta}_{k,i})}}{\sum_{l=1}^{K} [e^{-\frac{1}{\lambda} S(\tilde{\theta}_{l,i})}]}$
3) **For** $i = 1 \dots N$, compute: $[\delta\theta]_i = \sum_{k=1}^{K} P(\tilde{\theta}_{k,i})[\epsilon_k]_i$
4) Update $\theta \leftarrow \theta + \delta\theta$
5) Calculate cost: $Q(\theta) = \phi + \sum_{i=1}^{N} q(\theta_i)$
Execute first action, θ_1

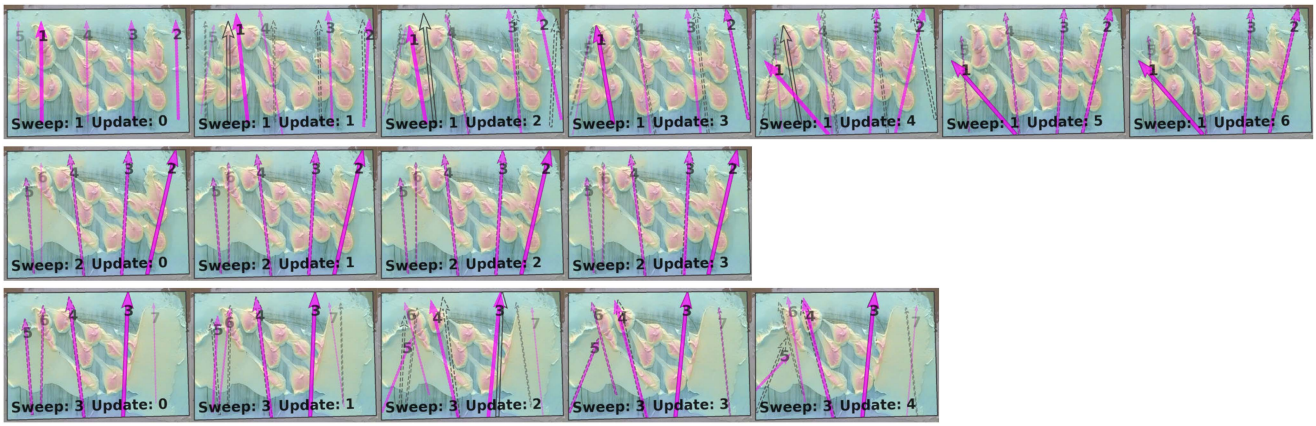


Fig. 4. Illustration of how the trowelling actions are updated. The magenta arrows indicate the updated actions and those in grey show the initialization based on the previous step. The actions are initialized for a horizon length of N (here 5) and iteratively updated until convergence. At this point, the first action is executed, and the process repeats. The optimisation is generally warmstarted using the previous solution. This cannot be done for the initial sweep and thus typically more updates are required for convergence (here 6) than for the remaining sweeps. The colored overlay shows the observed elevation map. The numbers on the arrows are their absolute indices from the start of the experiment. We show the first few sweeps illustrating the update process. For progress pictures including the final result see Fig. 6.

1) Cost: We define the total cost to optimize:

$$Q(\theta) = \phi + \sum_{i=1}^{N} q(\theta_i) \tag{4}$$

with N as the horizon length, ϕ the terminal cost defined as the sum of absolute errors between the target and predicted elevation at the end of the horizon, and $q(\theta_i)$ as the stage cost for a single action:

$$q(\theta_i) = \Delta V_{total} \cdot \beta_V + d \cdot \beta_d \tag{5}$$

ΔV_{total} penalizing actions that remove plaster volume from the overall system, i.e., material inside the work area or accumulated on the trowel. d penalizes longer actions to discourage unnecessarily passing over already completed areas. β_d and β_V are tuneable coefficients.

Where STOMP can omit the terminal cost and optimize only independent stage costs, we let the terminal cost be the dominating term that captures the progress toward the target surface. Furthermore, we sum subsequent stage costs as these can be affected by previous actions, e.g., filling up the accumulation buffer increases the likelihood of incurring costs with the following actions.

2) Receding Horizon: While STOMP optimizes over the whole task length, which is assumed fixed, we optimize over a fixed horizon length, N, and execute only the first action before re-optimizing in a typical receding horizon fashion. This helps accommodating model mismatches between the simulator and the real world.

3) Initialisation: The optimization must be initialized with a sequence of actions. For this, we evaluate two approaches; selecting actions uniformly at random, and using an informed initialization based on the strips heuristic. Due to the receding horizon approach, full sequence initialization is only necessary once. Hereafter, only the last action in the new sequence has to be explicitly initialized, as the remaining can be carried over. An example of how action sequences are updated using PLASTR is illustrated in Fig. 4.

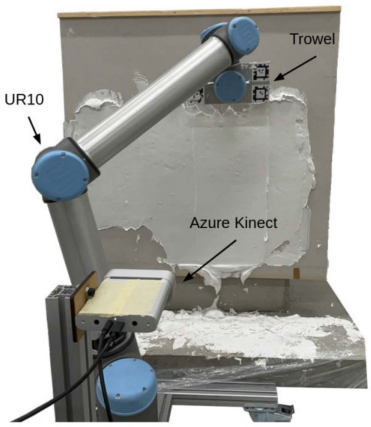
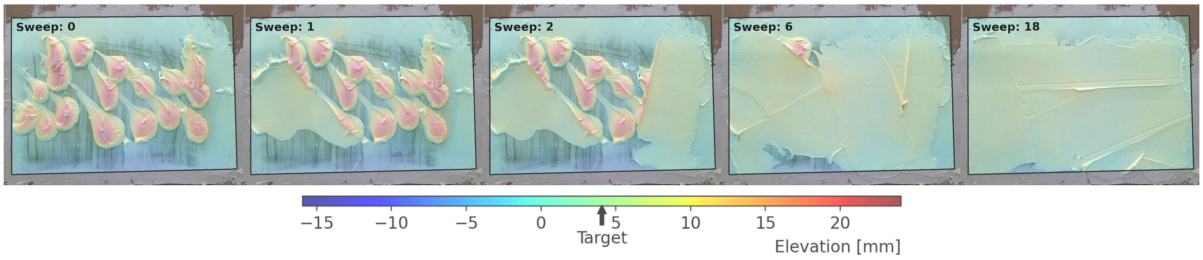


Fig. 5. The hardware setup constructed for physical experiments. The manipulator and camera are positioned in front of the work surface in the vertical setting.

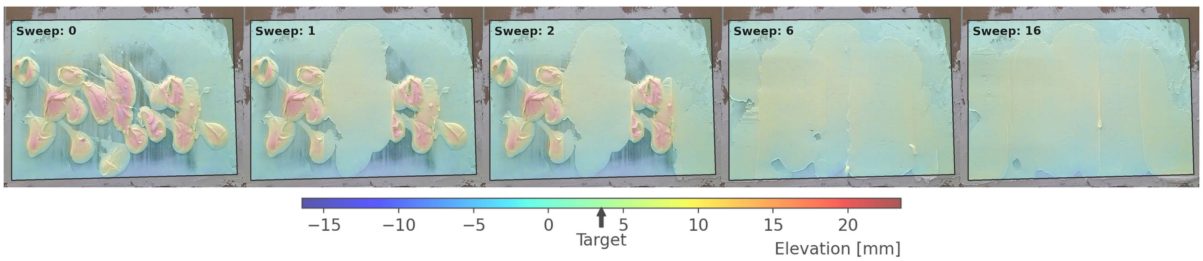
C. Heuristic-Based Baselines

As heuristics have been popular for planning in related literature, we propose two heuristics to evaluate as baselines. *High-to-low* is inspired by existing works [18], [19]. *Strips* is proposed by us specifically for plaster trowelling:

1) *High-to-low:* This approach places the start and end points at the highest and lowest points in the elevation map, respectively. It aims to accumulate and deposit in the same sweep.
2) *Strips:* The work area is divided into vertical sections. The planner alternates between actions along the center lines of the sections with the most excess and most missing material. It aims to accumulate material with one action and deposit with the next.



(a) *PLASTR* with strips initialisation ($N = 5$)



(b) *Strips* heuristic baseline

Fig. 6. Examples of trowelling progress for the optimization-based planner and the strips heuristic. The work area is indicated with a gray border, and the measured elevation map is overlaid. We observe in all experiments that spreading material fully to the edges of the work area poses a challenge. We attribute this to issues with establishing sufficient adhesion for reliable deposition which is addressed in Section V.

TABLE II
OVERVIEW OF EVALUATED PLANNING ALGORITHMS. *RANDOM* IS INCLUDED AS A SANITY CHECK AND *HIGH-TO-LOW* AND *STRIPS* ARE HERISTIC BASELINES. THE PLASTR ROW DEFINES DIFFERENT CONFIGURATIONS OF OUR PROPOSED OPTIMIZATION BASED PLANNER. REFER TO SECTION III FOR DETAILS

Random	Uniform sampling inside work area					
High-to-low	Action extension: 0.1 m at each end					
Strips	No. of sections: 20					
PLASTR	Initialisation	N	σ	K	β_V	β_d
'greedy'	Strips	1	0.1	25	2	2
'strips'	Strips	5	0.1	25	2	2
'random'	Random	5	0.1	25	2	2

IV. EXPERIMENTS

A. Setup

We use a UR10 robotic arm for hardware experiments. A statically mounted Azure Kinect provides depth measurements. The end-effector is a standard plastering trowel attached to the robot with 3D-printed parts. For future work, the mount allows for force-torque sensor integration. The experimental setup is shown in Fig. 5.

All experiments begin with the available plaster inside the work area as illustrated in the leftmost images in the progress sequences shown in Fig. 6. The work area is $0.6\ \mathrm{m}^2$ large and limited by the robot's reach. The target surface is determined at the beginning of each experiment as described in Section III-A, where ν is set as $100\ \mathrm{cm}^3$. The grid resolution, ρ, of the elevation map is fixed at $3\ \mathrm{mm}^2$. All actions are executed with a fixed linear velocity of $0.1\ \mathrm{m/s}$. In addition to the proposed planning algorithms, we include random actions as a sanity check. The evaluated methods are outlined with relevant parameters in Table II.

B. Performance Metrics

We evaluate performance using three metrics. To track how much plaster is lost, we use the normalized material volume:

$$V_{norm} = \frac{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\left[G_t(i,j)\right]}{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\left[G_0(i,j)\right]} \tag{6}$$

where G_0 refers to the initial elevation map and G_t the current, and m and n are the rows and columns of these. To evaluate how well the elevation matches the target surface, we propose the Root Mean Square Error (RMSE) as follows:

$$\mathrm{RMSE} = \sqrt{\frac{1}{m \cdot n} \cdot \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\left[\left(G(i,j) - T(i,j)\right)^2\right]} \tag{7}$$

where G and T are the actual and target surface grid maps. The third error metric, *completed area*, shows what percentage of the work area has the desired target height within a $2\ \mathrm{mm}$ error margin.

C. Simulation Results

The top part of Table III presents the results from the evaluated methods in simulation. The results are based on 52 different starting conditions collected from real-world data. The simulation experiments are terminated after 20 actions for all methods. In simulation, the optimization-based planner with random initialization achieves the best performance across all metrics, on average covering 99.8% of the work area. The *high-to-low* method has the worst performance covering only 92.6% on average, which is less than a 3 pp improvement from random actions. *Strips* performance is similar to the *PLASTR* planners in final result metrics. *PLASTR* is considerably more efficient than *Strips* as can be seen from the much faster convergence in Fig. 7 and the trowel distance results.

TABLE III
EXPERIMENT RESULTS FROM SIMULATION AND REAL WORLD

	Planner	Completed Area	V_{norm}	$RMSE$ [mm]	Trowel Distance [m]
Simulation	Random	0.898 ± 0.040	0.849 ± 0.041	1.367 ± 0.410	8.575 ± 0.768
	High-to-low	0.926 ± 0.067	0.879 ± 0.056	0.716 ± 0.510	13.529 ± 3.483
	Strips	0.994 ± 0.006	0.942 ± 0.011	0.277 ± 0.121	13.720 ± 0.095
	PLASTR 'greedy'	0.990 ± 0.011	0.937 ± 0.011	0.385 ± 0.196	4.302 ± 0.476
	PLASTR 'strips'	0.997 ± 0.002	0.945 ± 0.007	0.200 ± 0.084	5.421 ± 0.635
	PLASTR 'random'	$\mathbf{0.998 \pm 0.001}$	$\mathbf{0.946 \pm 0.007}$	$\mathbf{0.151 \pm 0.052}$	5.586 ± 0.604
Real	Random	0.573 ± 0.108	0.766 ± 0.074	2.626 ± 0.577	8.108 ± 1.734
	Strips	0.693 ± 0.038	0.734 ± 0.066	2.148 ± 0.113	10.049 ± 0.653
	PLASTR 'greedy'	0.657 ± 0.057	0.792 ± 0.060	$\mathbf{1.973 \pm 0.176}$	14.546 ± 1.540
	PLASTR 'strips'	0.682 ± 0.066	$\mathbf{0.824 \pm 0.028}$	1.997 ± 0.204	10.807 ± 1.169
	PLASTR 'random'	$\mathbf{0.702 \pm 0.046}$	0.781 ± 0.059	2.008 ± 0.234	11.475 ± 2.135

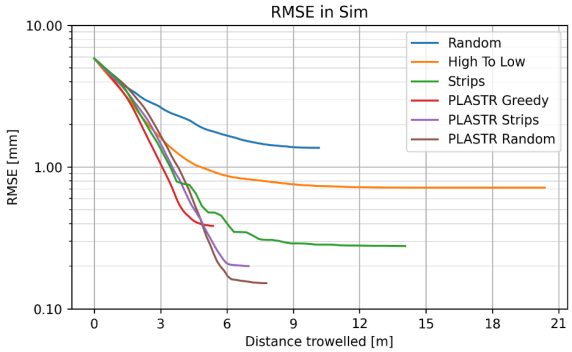


Fig. 7. From the RMSE relative to distance trowelled, it can be seen how PLASTR outperforms *strips* in simulation while converging roughly 50% faster. The RMSE is plotted on a logarithmic scale for visual clarity to distinguish the results.

D. Real-World Results

The accompanying video shows the results of the hardware experiments.[1] The bottom part of Table III shows the quantitative results of the experiments using the physical setup. *High-to-low* was left out due to poor simulation performance. For the remaining methods, each experiment was repeated five times, both for a vertical and a horizontal surface. Experiments were terminated when the results converged after a minimum of 15 actions. All experiments used the premixed Fixit 151 A plaster, which dries relatively slowly and thus can be reused. To avoid biases related to plaster drying out or other changes throughout the day, experiments were carried out in random order.

Compared to simulation, all methods lose more material. In particular, *strips* loses on average 26.6% of material. We observed that this was partly due to pushing the plaster outside the work area. Still, it also indicates that the accumulation and deposition processes are more complex than assumed in simulation. We also observe a large increase in RMSE for all methods. However, as shown previously in Table I, measurement noise alone may contribute up to $0.78\,\mathrm{mm}$ to the RMSE. This indicates that the measured results are close to the accuracy limits of the depth camera sensor. While no method outperforms the others in all three metrics, we see evidence that the optimization-based planner with random initialization performs the best in evenly spreading the plaster. It covers 70% of the work area within $\pm 2\,\mathrm{mm}$. The inconsistencies between simulation and real world indicates that measures should be taken to reduce the sim-to-real gap, as elaborated on in Section V. Distributing material to the edges of the work area has been observed to be challenging in all experiments which we attribute to insufficient adhesion control during deposition.

[1][Online]. Available: https://youtu.be/xlI1H1Hyx9w

V. CONCLUSIONS AND FUTURE WORK

We propose an optimization-based method for planning plaster trowelling, solving the task of spreading pre-applied plaster evenly across a surface.

In simulation, both the optimization-based planner and our newly proposed heuristic *strips* cover more than 99% of the work area with the desired tolerance. The baseline heuristic based on literature, *high-to-low*, barely improves upon random actions, showing the difficulty of designing effective heuristics. Our optimization-based planner offers a principled alternative to hand-crafted heuristics. It slightly outperforms *strips* in coverage while being 50% more efficient in conversion speed.

Our plastering robot can execute both the optimization-based and the *strips* plans in hardware experiments. The optimization-based planner outperforms the heuristics in the real world, too, but the overall performance drops compared to the simulation results. Parts of the performance decrease are caused by the limited accuracy of the Azure Kinect depth sensor. Purpose-build sensor setups could provide better feedback to the planners [33]. We, furthermore, observe that even in areas that are measured flat, with low RMSE, micro-imperfections like burrs can appear. In future work, such features could be detected in RGB images and treated in a finishing step. We identify the sim-to-real gap between the simulator and the real-world test setup as the major cause of performance decrease. To address this, the simulator could be improved by modeling the plaster interaction with greater detail, i.e. including side spill. More importantly, the lower-level execution of the planned troweling motion should be studied. When the robot moves the trowel along the target plane, the plaster does not always adhere to the surface. A force-controlled low-level controller could regulate the pressure of the plaster during the application process. This change would allow for better tracking of the desired deposition behavior modeled in the simulator, narrowing the sim-to-real gap.

The proposed planning strategy is not limited to the task of plaster smoothing. It could be adapted to manipulation planning for other viscous materials, such as artistic clay forming, earth moving with a bulldozer, or pizza dough handling in a kitchen environment.

ACKNOWLEDGMENT

We want to thank Giovanni Russo AG for their expertise shared in a hands-on plastering workshop and their support for the project. Furthermore, we thank Eliott Sounigo, Tsai Ping, and Dr. Selen Ercan for their valuable technical feedback and support.

REFERENCES

[1] M. Chui and J. Mischke, "The impact and opportunities of automation in construction," Dec. 2019. Accessed: Sep. 16, 2022. [Online]. Available: https://www.mckinsey.com/business-functions/operations/our-insights/the-impact-and-opportunities-of-automation-in-construction

[2] G. Pritschow, J. Kurz, J. Zeiher, S. McCormac, and M. Dalacker, "On-site mobile plastering robot: A practical design concept," in *Proc. IEEE/IAARC/IFAC 14th Int. Symp. Automat. Robot. Construction*, 1997, pp. 277–285.

[3] S. E. Jenny et al., "Robotic plaster spraying: Crafting surfaces with adaptive thin-layer printing," *3D Printing Additive Manuf.*, vol. 9, no. 3, pp. 177–188, 2022, doi: 10.1089/3dp.2020.0355.

[4] J. Forsberg, D. Graff, and Å. Wernersson, "An autonomous plastering robot for walls and ceilings," *IFAC Proc. Vol.*, vol. 28, no. 11, pp. 301–306, 1995. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667017469898

[5] J. Forsberg, R. Aarenstrup, and Å. Wernersson, "A construction robot for autonomous plastering of walls and ceilings," in *Proc. IEEE/IAARC/IFAC 14th Int. Symp. Automat. Robot. Construction*, 1997, pp. 260–268.

[6] A. Bulgakov, T. Bock, and J. Otto, "Robot manipulators for plastering work," in *Proc. Int. Multi- Conf. Ind. Eng. Modern Technol.*, 2019, pp. 1–5.

[7] A. Bulgakov and S. Emelianov, "Robotic system for plaster and finishing works on the construction site," in *Proc. Future Technol. Conf.*, 2016, pp. 885–889.

[8] T. Bock, N. Buzalo, and A. Bulgakov, "Mathematical description and optimization of robot control for plastering works," in *Proc. Int. Multi-Conf. Ind. Eng. Modern Technol.*, 2018, pp. 1–5.

[9] T. Long, E. Li, Z. Fang, W. Zhao, and Z. Liang, "A novel measurement and control method for automatic plastering machine," in *Proc. IEEE Int. Conf. Mechatron. Automat.*, 2015, pp. 2325–2330.

[10] Y. Liu and H. Yu, "A positioning method of intelligent plastering robot head," in *Proc. 13th Int. Symp. Comput. Intell. Des.*, 2020, pp. 318–321.

[11] Z. Liu, D. Chen, X. Jiang, and Y. Liu, "Putty plastering realized by a force controlled robotic scraper," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2021, pp. 1034–1039.

[12] S. E. Jenny, E. Lloret-Fritschi, F. Gramazio, and M. Kohler, "Crafting plaster through continuous mobile robotic fabrication on-site," *Construction Robot.*, vol. 4, no. 3, pp. 261–271, Dec. 2020, doi: 10.1007/s41693-020-00043-8.

[13] S. E. Jenny et al., "Continuous mobile thin-layer on-site printing," *Automat. Construction*, vol. 146, 2023, Art. no. 104634. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0926580522005040

[14] D. Mitterberger et al., "Interactive robotic plastering: Augmented interactive design and fabrication for on-site robotic plastering," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, Apr. 2022, Art. no. 174, doi: 10.1145/3491102.3501842.

[15] X. Jiang and X. Li, "Robotized interior finishing operations with visual feedback," *Ind. Robot: Int. J. Robot. Res. Appl.*, vol. 49, no. 1, pp. 141–149, Jan. 2022, doi: 10.1108/IR-02-2021-0034.

[16] X. Li and X. Jiang, "Development of a robot system for applying putty on plastered walls," in *Proc. IEEE Int. Conf. Mechatron. Automat.*, 2018, pp. 1417–1422.

[17] F. Raspall, F. Amtsberg, and S. Peters, *Mater. Feedback in Robotic Prod.*. Berlin, Germany: Springer, 2014, pp. 333–345. doi: 10.1007/978-3-319-04663-1_23.

[18] J. Ondras, D. Ni, X. Deng, Z. Gu, H. Zheng, and T. Bhattacharjee, "Robotic dough shaping," in *Proc. 22nd Int. Conf. Control, Automat. Syst.*, 2022, pp. 300–307, doi: 10.23919/ICCAS55662.2022.10003767.

[19] D. Jud, I. Hurkxkens, C. Girot, and M. Hutter, "Robotic embankment," *Construction Robot.*, vol. 5, no. 2, pp. 101–113, Jun. 2021.

[20] A. Cherubini, V. Ortenzi, A. Cosgun, R. Lee, and P. Corke, "Model-free vision-based shaping of deformable plastic materials," *Int. J. Robot. Res.*, vol. 39, no. 14, pp. 1739–1759, 2020. doi: 10.1177/0278364920907684.

[21] C. Schenck, J. Tompson, S. Levine, and D. Fox, "Learning robotic manipulation of granular media," in *Proc. 1st Annu. Conf. Robot Learn.*, 2017, pp. 239–248. [Online]. Available: https://proceedings.mlr.press/v78/schenck17a.html

[22] Y. Zhang, W. Yu, C. K. Liu, C. Kemp, and G. Turk, "Learning to manipulate amorphous materials," *ACM Trans. Graph.*, vol. 39, no. 6, Nov. 2020, Art. no. 189. doi: 10.1145/3414685.3417868.

[23] D. Seita et al., "Learning to rearrange deformable cables, fabrics, and bags with goal-conditioned transporter networks," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 4568–4575.

[24] Y. Li, J. Wu, R. Tedrake, J. B. Tenenbaum, and A. Torralba, "Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids," in *Proc. 7th Int. Conf. Learn. Representations*, New Orleans, LA, USA, May 6-9, 2019. [Online]. Available: https://dblp.org/rec/conf/iclr/LiWTTT19.html?view=bibtex

[25] N. Tuomainen, D. Blanco-Mulero, and V. Kyrki, "Manipulation of granular materials by learning particle interactions," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 5663–5670, Apr. 2022.

[26] E. Theodorou, J. Buchli, and S. Schaal, "Reinforcement learning of motor skills in high dimensions: A path integral approach," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2010, pp. 2397–2403.

[27] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 4569–4574.

[28] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim, "Unified particle physics for real-time applications," *ACM Trans. Graph.*, vol. 33, no. 4, Jul. 2014. [Online]. Available: https://doi.org/10.1145/2601097.2601152

[29] Y. Hu et al., "A moving least squares material point method with displacement discontinuity and two-way rigid body coupling," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–14, 2018.

[30] H. Barreiro, I. Garcıa-Fernández, I. Alduán, and M. A. Otaduy, "Conformation constraints for efficient viscoelastic fluid simulation," *ACM Trans. Graph.*, vol. 36, no. 6, Nov. 2017, Art. no. 221. [Online]. Available: https://doi.org/10.1145/3130800.3130854

[31] P. Fankhauser and M. Hutter, "A universal grid map library: Implementation and use case for rough terrain navigation," in *Robot Operating System (ROS) – The Complete Reference (Volume 1)*, A. Koubaa, ed. Berlin, Germany: Springer, 2016, ch. 5, pp. 99–120. [Online]. Available: http://www.springer.com/de/book/9783319260525

[32] T. Miki, L. Wellhausen, R. Grandia, F. Jenelten, T. Homberger, and M. Hutter, "Elevation mapping for locomotion and navigation using GPU," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 2273–2280.

[33] T. F. Lam, H. Blum, R. Siegwart, and A. Gawel, "Sl sensor: An open-source, real-time and robot operating system-based structured light sensor for high accuracy construction robotic applications," *Automat. Construction*, vol. 142, 2022, Art. no. 104424. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0926580522002977