This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2846278, IEEE Robotics and Automation Letters

PaintCopter: An Autonomous UAV for Spray Painting on 3D Surfaces

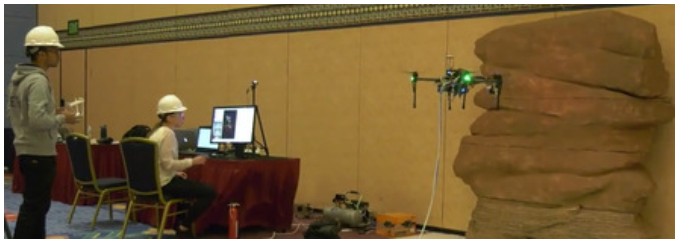
Anurag Sai Vempati[1,2], Mina Kamel[1], Nikola Stilinovic[2], Qixuan Zhang[2], Dorothea Reusser[2]
Inkyu Sa[1], Juan Nieto[1], Roland Siegwart[1] and Paul Beardsley[2]

***Abstract*—This paper describes a system for autonomous spray painting using a UAV, suitable for industrial applications. The work is motivated by the potential for such a system to achieve accurate and fast painting results. The PaintCopter is a quadrotor that has been custom fitted with an arm plus a spray gun on a pan-tilt mechanism. To enable long deployment times for industrial painting tasks, power and paint are delivered by lines from an external unit. The ability to paint planar surfaces such as walls in single color is a basic requirement for a spray painting system. But this work addresses more sophisticated operation that subsumes the basic task, including painting on 3D structure, and painting of a desired texture appearance. System operation consists of (a) an offline component to capture a 3D model of the target surface, (b) an offline component to design the painted surface appearance, and generate the associated robotic painting commands, (c) a live system that carries out the spray painting. Experimental results demonstrate autonomous spray painting by the UAV, doing area fill and versatile line painting on a 3D surface.**

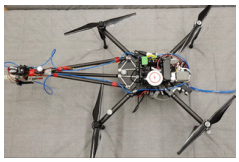
***Index Terms*—Aerial Systems: Applications; Computer Vision for Automation; Industrial Robots**

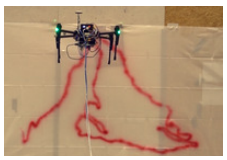
I. Introduction

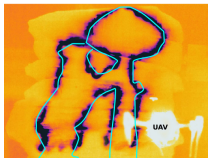
ROBOTIC painting using a UAV has the potential to produce accurate (predictable and repeatable) painted appearance, to be low-cost, and to avoid the need for scaffolding and ladders. This motivates our work on PaintCopter for autonomous spray painting. A basic painting task is to paint a planar surface such as a wall in a single color. This paper attacks a larger challenge that subsumes the more basic task in two ways - with the ability to paint on 3D structure, and to paint texture. Fig. 1 provides an illustration in which painting is being done on a synthetic rock, and the goal is to paint a uniform base color and then to overlay color striations, in order to produce a (user-designed) rock-like surface appearance. This type of task - with the two dimensions of painting on a 3D object and painting a texture for theming/styling - is the motivation for our work. It requires a more sophisticated approach than the single color wall painting problem, and explains design choices that might have been avoided with more basic functionality - for example, the use of a spray gun on a pan-tilt unit (PTU) instead of on a rigid mount.



(a)







(b) (c) (d)

Fig. 1: **(a)-(b)** The PaintCopter is a modified commercial quadrotor UAV. An arm holds a spray gun on a pan-tilt unit, outside of the reach of rotor wash. Power and paint are supplied from an external source, to enable unconstrained flight time. Spray painting experiments are done either with paint on a canvas **(c)** or with water **(d)**, in which case the spray pattern is visualized with a thermal camera.

This paper describes a fully integrated pipeline from 3D capture of a target structure, through a user's design of the desired surface appearance, through to a live autonomous system with two modes of spray painting - area-fill and line painting. The contributions of the work are (a) the first demonstration to our knowledge of an autonomous UAV for spray painting, and (b) a complete case study with an exposition of application challenges, adopted solutions, and quantitative and qualitative experimental results.

In the remainder, Section II describes related work, Section III describes design and hardware, Section IV covers software architecture, Section V covers control of the UAV and spray gun, and Section VI describes the full painting pipeline as a multi-stage mission. Section VII provides experimental results and finally we conclude with some remarks in Section VIII.

II. Related Work

Painting is one of the earliest applications of industrial robots. Several systems have been proposed to achieve de-

Manuscript received: February, 24, 2018; Revised May, 15, 2018; Accepted June, 5, 2018.

This paper was recommended for publication by Editor Jonathan Roberts upon evaluation of the Associate Editor and Reviewers' comments.

[1]Anurag Sai Vempati, Mina Kamel, Inkyu Sa, Juan Nieto and Roland Siegwart are with the Autonomous Systems Lab at ETH Zurich, Leonhardstrasse 21, 8092, Zurich, Switzerland. avempati@ethz.ch

[2]Anurag Sai Vempati, Nikola Stilinovic, Qixuan Zhang, Dorothea Reusser and Paul Beardsley are with Disney Research Zurich, Stampfenbachstrasse 48, 8006, Zurich, Switzerland. anurag.vempati@disneyresearch.com

Digital Object Identifier (DOI): see top of this page.

2377-3766 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2846278, IEEE Robotics and Automation Letters

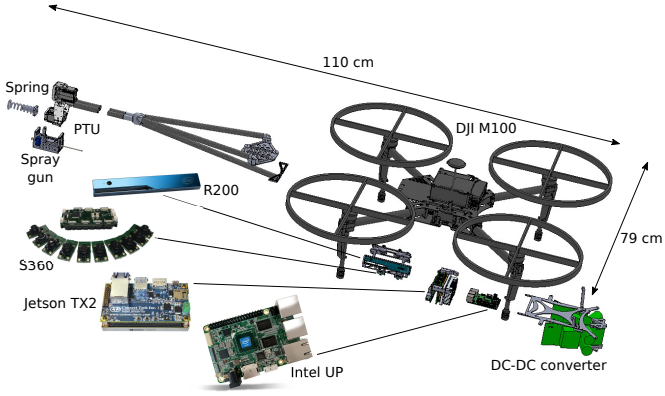


Fig. 2: PaintCopter and its hardware components

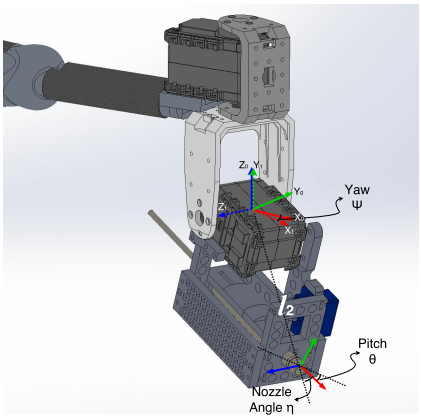


Fig. 3: Spray gun mounted on a Pan-Tilt Unit (PTU)

sired spray patterns on complex surfaces. Among the early works on trajectory planning for spray painting applications is the work of Suh [1]. The proposed method is iterative and takes into account the geometry of the object to be painted, coating uniformity and painting time. The approach has been demonstrated on planar surfaces. In [2] an improved trajectory generation framework for spray painting is presented that takes into account the tool model, paint distribution and coating uniformity. The approach has been validated on various curved surfaces. A method to achieve photorealistic gray-scale images on large planar surfaces has been presented in [3]. The algorithm generates tool-path based on image segmentation and a timing algorithm computes tool speed at each path point to avoid oversaturation. Handheld sprayers are combined with automatic control for the paint delivery in [4], [5]. These proposed systems mainly focused on coverage coating for automotive industry using industrial manipulators in structured environments.

UAVs are growing increasingly prominent within the precision agriculture domain for applications such as pesticide spraying. Commercial products such as DJI MG-1S[1] are already in use and research systems for efficient pesticide delivery [6] are being developed. Using UAVs for painting is relatively new. In [7] a method to create stippling prints using quadrotor UAV was presented. Their method exclusively used external motion capture system and was demonstrated only on planar canvas for short flight durations. In [8] the authors improved the system by supplying power via tether, thus being able to paint for extended durations. However, no painting delivery method was proposed and as a result the UAV has to regularly refill the ink by dabbing on a sponge filled with the ink. The work presented in this paper aims to achieve complex spray patterns on 3D surfaces for extended periods and it is not limited to working under motion capture systems.

III. Hardware Description

The PaintCopter and its hardware components are shown in Fig. 2. It has the following components: (a) A modified quadrotor platform with a custom arm and spray gun mounted on a pan-tilt unit, (b) Onboard electronics - a sensor rig and two computers, and (c) A tether composed of a power line and a paint line, leading to offboard power and paint sources.

A. Platform

The UAV hardware consists of a DJI Matrice 100[2] that is augmented with a custom arm and a spray gun mounted on a pan-tilt unit (PTU), as shown in Fig. 3. The arm consists of three carbon fiber tubes in a triangular configuration, with 3D printed aluminum mounts, and a total weight of 140g. The PTU has two Dynamixel[3] AX-12A servo motors to allow yaw $\pm 90°$ and pitch $\pm 45°$ control of the spray gun. An additional servo actuates the spray release, with controllable aperture between closed and open positions. A compliant spring - as depicted in Fig. 2 - is placed at the end of the arm beyond the spray gun, to prevent damage in case of accidental contact with the painting surface. The spring is located above the center-of-mass with the effect that, in case of contact with a surface, the UAV is pushed backwards in a predictable way (rear-end tip) with no loss of control. The onboard electronics have been shifted as far as possible to the opposite side of the UAV from the arm, to counter-balance the weight on the arm. As a result, the center-of-mass remains close (40mm) to the center axis of the platform. The total weight of the modified UAV is 4200g.

The UAV is tethered to provide a continuous supply of power and paint, to avoid constraints on deployment time. The tether consists of a paint line and compressed air line, a power line, and an ethernet cable for data streaming, and arrives from above the UAV. A rotor cage prevents the tether from coming into contact with the propellers. The length of the tether can be adjusted depending on the size of the UAV's workspace. Having the tether arrive from above or below the UAV is an application-dependent choice. For a tether that arrives from below, the drag limits maximum altitude. For a tether that arrives from above, drag is avoided but the external power and paint units must be above the painting surface, say on the roof of a building or on a cherry picker that is positioned above the painting area. Though we tested both choices, for our small scale experiments we work mainly with the first choice since the drag is minimal.

[1] www.dji.com/mg-1s

[2] https://www.dji.com/matrice100

[3] http://www.robotis.us/dynamixel/

2377-3766 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2846278, IEEE Robotics and Automation Letters

B. Onboard Electronics

The onboard sensor rig consists of a S360 stereo camera[4] sensor, an ADIS IMU[5], and an Intel RealSense R200 depth camera[6]. The sensors are calibrated against each other using Kalibr [9]. The S360 is used for visual-inertial odometry, while the RealSense offers depth and color information.

PaintCopter has two computers for performing onboard operations. An Intel UP board[7] is used for flight control software and processing visual-inertial odometry. A Jetson TX2 board[8] is used for performing GPU processing involving dense 3D mapping and localization. It is also used to direct the spray gun (more detail in Section V-C). Both these boards are communicating via ethernet with a base station for high-level inputs via a graphical user interface (GUI). *Note:* All processing is performed onboard to avoid unexpected behavior in cases of communication loss or time delays.

C. Offboard Electronics

Offboard power is realized using a ground unit (1600W 220AC/400DC step-up converter) and onboard unit (2400W 400DC/24DC step-down converter). The UAV battery was replaced by a custom high voltage DC/DC converter, along with associated modification to the communication protocol to cater for the modification. The advantage of a high voltage tether between the offboard unit and the platform is that it can be realized using thin wires (22-24 AWG). The offboard paint unit consists of an air compressor operating at 1-4 bar, and a paint reservoir.

IV. Software Architecture

The software runs on three computers - the Intel UP board, Jetson TX2, and a base station. All processes critical to the UAV operation are executed onboard and not on the base station, so that communication failure or time delays cannot cause a malfunction of the UAV operation. Fig. 4 shows an overview of the software packages. In the rest of this section, we briefly describe each module integrated or developed in our system.

Modules involved in autonomous flight are executed on the Intel UP board. The **S360 driver** module provides the grayscale images and IMU data from the S360 camera. **ROVIO** [10] performs visual-inertial state estimation - inertial measurements from the IMU are used for filter propagation while multi-level feature patches in the image are tracked for performing filter updates. The **Pose Sensor Fusion** [11] module uses an Extended Kalman Filter (EKF) to further fuse the pose estimates from ROVIO with the UAV's IMU data, to give high frequency odometry estimates. A ROS interface for the DJI platform provides an interface for the onboard autopilot developed by the manufacturer. The **Position Controller** is based on non-linear model predictive control [12] and uses the odometry data together with position commands to provide attitude commands to the DJI autopilot. The controller is explained in more detail in Section V.

Modules related to the **PTU control** and **SLAM** modules are executed on the Jetson TX2 platform. The **R200 driver** module provides the depth and RGB images from Realsense camera. We integrated the SLAM system from our previous work [13], which uses odometry data from the **Pose Sensor Fusion** module together with the depth and RGB images to provide accurate UAV pose. Modifications made to the SLAM system are detailed in Section VI. The **PTU driver** module provides ROS interface to the dynamixel motors and the servo. The **PTU control** module uses the pose estimates from the **Pose Sensor Fusion** module along with the target spray points on the painting surface to provide motor commands to the PTU.

The Base Station is a Linux machine running the GUI for launching different processes on both the onboard computers. The **GUI** is also used to visualize data streams from the various sensors and to monitor the core processes, such as SLAM and UAV control. The user can send high-level commands to the UAV, for example, to initiate autonomous flight mode, to provide a painting command sequence etc. The **Relocalization Node** aligns the map from the current mission with the previously scanned map of the environment, to ensure a consistent global frame of reference.

V. UAV Control

A. Trajectory Tracking Controller

A model-based predictive controller [12] is used to follow the reference trajectory generated by the task planning module. The controller is based on a cascade scheme, where a high level trajectory tracking controller is generating attitude commands to be followed by the low level attitude controller running on the platform autopilot. The trajectory tracking controller takes into consideration the physical limitations of the platform and compensates for external disturbances such as wind and forces due to the power and paint lines as described in [12], [14].

B. System Identification

The behaviour of the inner attitude loop is taken into account in the trajectory tracking controller. To this end, a system identification procedure is followed [15]. We apply an excitation input of the platform in the form of attitude commands and the system responses are recorded. A first order model of the system behaviour is derived following frequency domain system identification techniques.

C. Spray Gun Control

Given that the UAV is in vicinity of the desired target location, the pan-tilt motors have to be controlled such that the spray gun is always pointing in the direction of a given target spray point in space. Now, given a target spray point $P^G_{ref} = [P^G_{ref,x}, P^G_{ref,y}, P^G_{ref,z}]$ expressed in a global reference

[4]http://www.zurichsense.com
[5]http://www.analog.com
[6]https://software.intel.com/en-us/realsense
[7]http://www.up-board.org/
[8]https://developer.nvidia.com/embedded-computing

2377-3766 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2846278, IEEE Robotics and Automation Letters

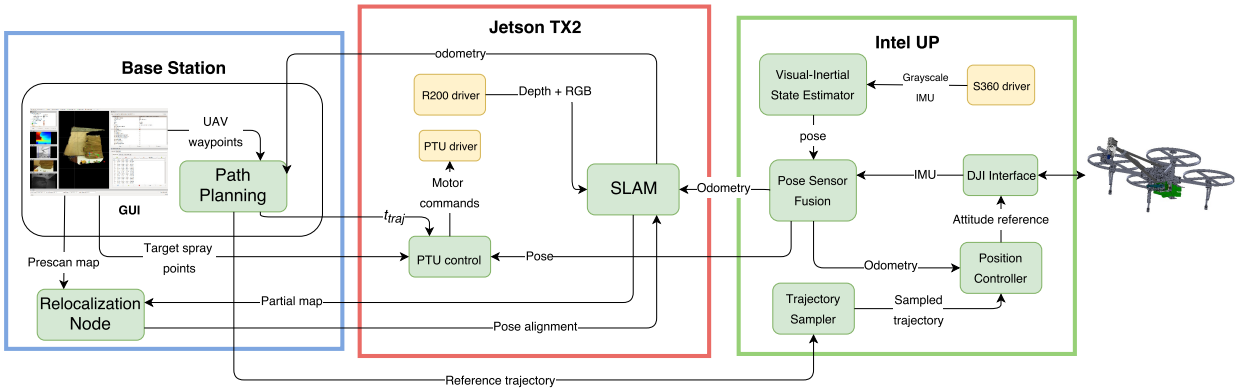


Fig. 4: Flowchart indicating individual packages and data transfers.

frame G, yaw ψ_{ref} and pitch θ_{ref} commands for the PTU are computed as follows:

$$
\begin{aligned}
P^0_{ref} &= T^0_{body} T^{body}_G P^G_{ref} \\
\psi_{ref} &= \tan^{-1}\left(\frac{P^0_{ref,y}}{P^0_{ref,x}}\right) \\
P^1_{ref} &= \begin{pmatrix} \cos\psi & \sin\psi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \sin\psi & -\cos\psi & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} P^0_{ref} \\
p_1 &= \tan^{-1}\left(\frac{P^1_{ref,y}}{P^1_{ref,x}}\right) \\
d &= \sqrt{\|P^1_{ref}\|^2 - l_2^2 \sin^2\eta} - l_2\cos\eta \\
p_2 &= \tan^{-1}\left(\frac{d\sin\eta}{l_2 + d\cos\eta}\right) \\
\theta_{ref} &= p_1 - p_2 + \eta
\end{aligned}
\tag{1}
$$

where, T^{body}_G is the pose of the UAV's body frame in G and T^0_{body} and η are hardware design parameters. Please refer to Fig. 3 for coordinate system convention and other notation.

Furthermore, the servo actuating the spray gun's nozzle is opened only when the spray gun is within a defined distance from the target spray point. See more on the spray gun's operational range in Section VII. Modulating the paint release between barely open to fully open aperture depending on the distance to the rock was found to have barely noticeable effect on the end result so it's always fully open when in operational range from the target spray point.

VI. Painting a Surface

The proposed painting pipeline has been designed as a multi-session process. First the target surface is scanned and a Truncated Signed Distance Field (TSDF) plus associated data (mesh, hash table) is stored to disk[9]. Next is an offline stage, in which a user can specify a desired surface design, using the 3D model visualized in a GUI. Note that the user is specifying appearance only, and not anything to do with UAV control. The desired surface design is used to automatically compute waypoints for the UAV, and target spray points on the surface for the painting. Finally in an online painting session, the UAV uses the stored 3D model to re-localize itself against the target surface, and follows the trajectory defined by the waypoints. The spray gun is directed according to the target spray points on the surface.

[9]The term '3D model' is used in the remainder to refer to this data.

A. Scanning the Target Surface

The target surface is scanned using the method presented in [13] with the UAV under manual control, as shown in Fig. 5a. The scan generates a TSDF which is stored on disk with an associated mesh and hash table. The scanning framework is a GPU parallelized SLAM system capable of using photometric data, depth data, and visual-inertial odometry from an active RGB-D sensor to build accurate dense 3D maps of indoor environments at high frame-rate. The system has been designed to be robust to abrupt camera motion typically observed on dynamic systems such as a UAV. A hybrid CPU-GPU framework enables scanning of large scale environments that exceed GPU memory alone.

B. Specifying the Surface Design, and Task Planning

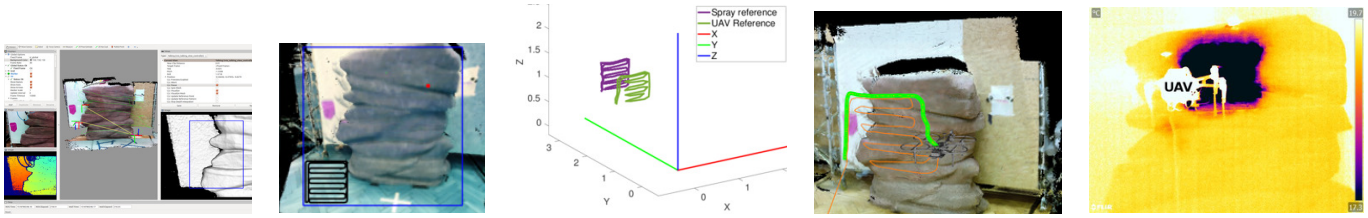
The stored 3D model is used in two ways - firstly to enable a user to specify the surface design for the painted surface, and secondly as a basis for generating meta-data that defines the painting mission - the UAV waypoints and the target spray points on the surface.

The generation of painting commands for arbitrary textures, which would include effects like color gradients, is out of scope for this work. Here we address two types of painting - area fill and line painting - that are sufficient for painting a base color with overlaid striations. Area fill is currently based directly on the line painting, with the user manually specifying a lawn mowing line pattern to cover the desired area of the target surface (although this is clearly amenable to be automated). Line painting is specified in a GUI that shows a visualization of the stored 3D model, and allows the user to specify an arbitrary line for painting, as shown in Fig. 5b.

The desired painting is first split into set of continuous line segments and a painting mission is generated for each individual segment. Each line segment on the target surface defines the target spray points for the spray gun control described

2377-3766 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2846278, IEEE Robotics and Automation Letters



(a) Target surface is scanned and a 3D model is stored to disk

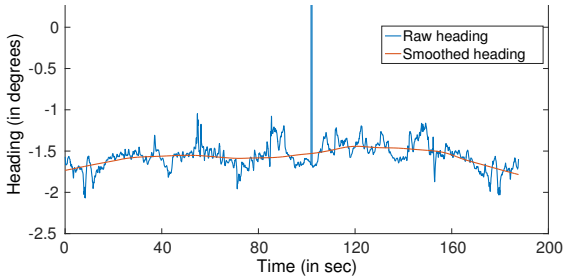
(b) A GUI-based tool allows a user to specify the surface design on the 3D model

(c) UAV waypoints plus target spray points on the surface, automatically generated during task planning

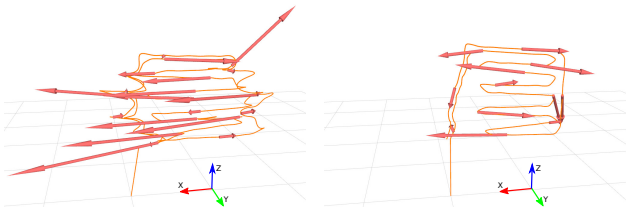
(d) The UAV follows the specified waypoints by localizing against the 3D model

(e) The PTU is controlled to direct spray at the target spray points on the surface

Fig. 5: Steps in specifying and doing area fill painting for a square region on a target surface.



(a) Pre-smoothed and smoothed headings



(b) Trajectory generated using pre-smoothed headings

(c) Trajectory generated using smoothed headings

Fig. 6: Effect of heading direction smoothing in generating the UAV trajectory. *Note*: Using pre-smoothed heading directions results in a UAV trajectory with crossovers and sharp turns, which result in high UAV accelerations (red arrows) (Fig. 6b). In contrast, the trajectory generated from smoothed heading directions has no crossovers or high accelerations, and is a much smoother trajectory (Fig. 6c).

in Section V-C. The preliminary waypoints for the UAV are generated as follows. For each target spray point, its surface normal is projected onto the ground plane. A corresponding waypoint for the UAV is generated at a fixed offset from the spray point along its projected surface normal, while the orientation of the UAV points towards the target spray point. The preliminary waypoints are then smoothed using a LOWESS filter [16] by fitting a locally-weighted polynomial regression, and a final set of waypoints are generated (Fig. 5c). As seen in Fig. 6 , this smoothing avoids crossovers and high accelerations in the UAV trajectory arising from areas of high curvature on the target surface.

The mesh of the scanned target surface is also used to generate feature descriptors that are used for relocalization in the painting session. In summary, the data for the target surface after this stage of processing consists of the TSDF, mesh, hash table, UAV waypoints, target spray points on the surface, and feature descriptors for the surface.

C. Painting Session

The flowchart for the online system is shown in Fig. 7. The three components of the painting session are (a) initialising the coordinate frame at startup, (b) waypoint traversal (c) the painting itself.

1) Initialising the coordinate frame at startup: To be able to use the 3D model plus associated meta-data such as waypoints and target spray points obtained from the task planning stage, we need a way to operate the UAV in the same global reference frame as that of the pre-scan session. So, on startup, the coordinate frame of the UAV is initialised by aligning it to the coordinate frame of the stored 3D model as follows. The TSDF and hash table of the stored 3D model are loaded. The UAV scans the target surface for a few seconds using the SLAM system, in an arbitrary coordinate frame determined by its start location. The scanning is performed in reserve memory in the GPU, and does not affect the loaded 3D model. A TSDF is generated for the live view of the target surface which is used to generate a point cloud with associated normals, and the **Relocalization Node** computes the transformation between the live point cloud and the stored 3D model.

For relocalization, global alignment between the live point cloud and the stored 3D model is achieved using fast global registration [17].[10]. ICP is then performed between the two point clouds to refine the transformation. The computed transformation is supplied to the *Camera Tracking* module of the SLAM system as a pose alignment correction, thus putting the SLAM system into the same coordinate frame as the stored 3D model. The transient scene data in reserve GPU memory is discarded, and the SLAM system proceeds according to its normal operation but with *Depth Fusion* turned-off (since the 3D model is not being updated).

2) Waypoint traversal: Waypoints for the painting mission are part of the meta-data computed earlier during the task planning. These waypoints are input to the **Path Planning** module, which performs polynomial trajectory optimization to generate feasible polynomial trajectory segments for the UAV

[10]There are alternative ways to realise this stage in our application, for example the UAV may be starting from an approximately known position, which is a practical option that can be encoded in the meta-data for the stored 3D model.

2377-3766 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2846278, IEEE Robotics and Automation Letters

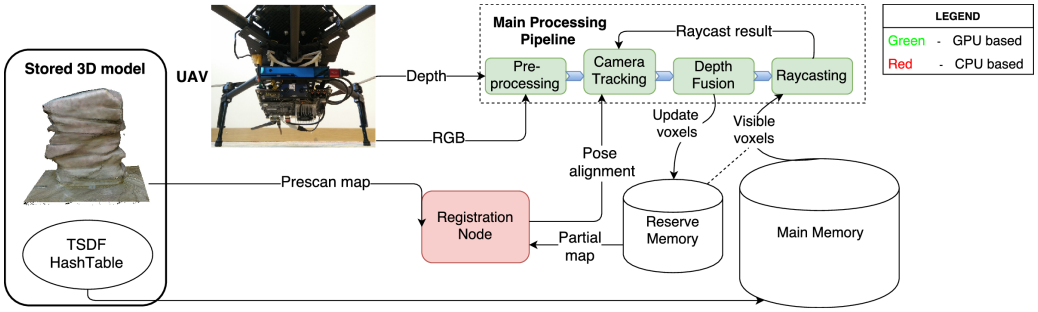


Fig. 7: Flowchart indicating individual modules and data transfers involved in the online SLAM pipeline. Only showing relevant modules. Please refer to [13] for a full description.

. The implementation is based on [18] with extensions to the non-linear optimization described in [19]. The optimization is performed such that the snap space of the polynomial segments is continuous, while respecting the constraints on maximum UAV velocity and acceleration. After receiving a high level input from the base station to initiate painting, these polynomial segments are sampled by the **Trajectory Sampler** module to obtain feasible reference waypoints for the **Position Controller** (Fig. 5d).

3) Painting: During waypoint traversal, the direction of the spray gun is adjusted using the PTU to direct paint at the target spray points on the surface, as described in Section V-C (Fig. 5e). The UAV motion is coupled to the control of the PTU by defining a velocity $v_{ref} = \wp/t_{traj}$, where $\wp$ is the total length of the painting curve on the surface, and t_{traj} is the estimated time for executing the full trajectory, as obtained from the **Path Planning** module. Iterating through the target spray points at v_{ref} ensures that the drone trajectory and painting proceed in a coupled way and terminate at the same time, assuming that the UAV is moving at uniform speed. The latter assumption is not guaranteed but the PTU has a large workspace, and the system is able to compensate for deviations in the UAV's speed from the average speed.

VII. Experimental Results

The system has been tested in simulation and in real-world indoor experiments. Simulation tests were performed using the RotorS [20] gazebo simulation platform. Indoor tests used either the Vicon motion capture system[11] or the SLAM system for localization - in the remainder, it is stated explicitly when SLAM is used. At take-off, AprilTag fiducials [21] viewed by a downward-facing camera are used to supplement the SLAM system. This was needed to remove accumulated drift during take-off due to drastic scale changes in the downward facing camera. This is special-purpose, used for robustness at launch only. Most of the paintings are done with water instead of paint. A FLIR thermal camera[12] is used to show the spray pattern on the surface. The work of this paper can be illustrated with water spraying, and that is much more convenient to use. Nevertheless, in Section VII-D, we provide a result when actual paint is used. Outdoor experiments are future work - items to be investigated are the effects of wind, and the effect of a longer tether length.

In the remainder of the section, quantitative and qualitative experimental results are provided for area fill and for line painting. Various representational designs are painted to demonstrate line painting - this differs from our application goal which is to paint abstract themed/styled texture, however the representational line painting aids evaluation.

[11]https://www.vicon.com/

[12]https://www.flir.com/

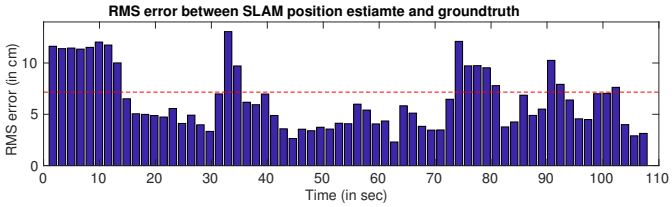


Fig. 8: SLAM position estimates as compared with ground truth. The average RMS error (in red) is about 7cm.

A. Tracking Accuracy

During painting, the spray gun is positioned in the 5-15cm range from the rock surface. For line segments to have uniform line thickness on the surface, a constant distance (about 10cm) of the spray gun from the surface is maintained.

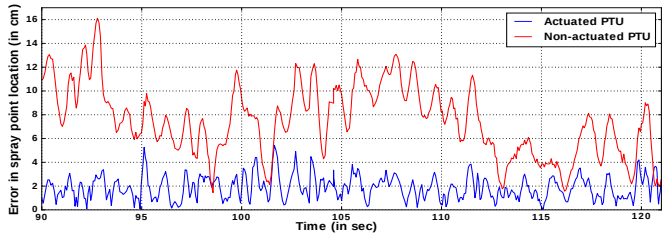
The quality of the painting result depends on the accuracy of camera tracking, and a comparison of the pose estimates of the SLAM system with ground truth from the Vicon motion capture system is shown in Fig. 8. The trajectory is more than 150m in total length and is several minutes long with several sharp accelerations. The total Root-Mean-Square (RMS) error is around 7cm, indicating that the SLAM tracking accuracy is precise enough for the UAV to operate in close proximity with the target surface.

B. Painting Accuracy

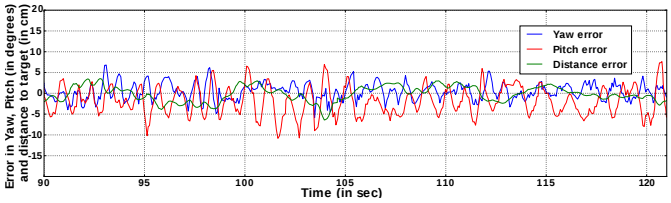
Three different metrics are used to evaluate the accuracy of a painted pattern. For the first metric, the location of the physical spray pattern on the surface is inferred by intersecting the spray gun direction with the 3D model of the surface throughout the mission. (The nozzle produces a conical spray, and the axis of the cone is intersected with the 3D model). This

2377-3766 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2846278, IEEE Robotics and Automation Letters



(a) Error in spray location



(b) Errors in yaw, pitch and distance to target surface

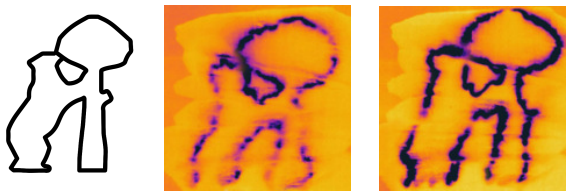
Fig. 9: Results for painting accuracy during line painting.

inferred spray pattern is compared with the pre-specified spray pattern, and a plot of error over time is shown in Fig. 9a. This is a typical experiment, and the error in the spray location is almost always within 4cm and the RMS error is 2.12cm. The second metric is the error in actual yaw and pitch of the spray gun during the experiment compared with the desired yaw and pitch at each time instant. This is shown in Fig. 9b. The error in both yaw and pitch is almost always within $\pm 5°$, with the RMS error of 2.05°and 3.94°in yaw and pitch respectively. Due to the conical nature of the spray, to get uniformly thick line paintings, it's crucial that the nozzle stays at a fixed distance from the target surface. So, as a third metric, we compare the error in nozzle's distance to the target surface. This is shown in Fig. 9b. The error in distance from the surface is almost always within ± 5cm, with a standard deviation of 1.88cm.

The PTU is needed for painting in concavities, in which case the system aligns the spray direction as nearly as possible with the surface normal. The PTU also allows dexterity in doing painting without needing to do aggressive UAV maneuvers. To evaluate the contribution of the PTU, the spraying accuracy is compared with and without an actuated PTU. Fig. 9a shows that spraying accuracy degrades by almost 4 times when the PTU is not actuated. Further, one can see jumps in error when using a non-actuated PTU (between 90-95 and 105-110 s), which is not the case when using an actuated PTU. This happens when the UAV passes through curved regions on the target surface where the need for an actuated PTU is more evident.

C. Need for a Dense 3D Model

This section describes how having a dense 3D model of the target surface is essential to painting quality. During *Task Planning*, the 3D model is used in the generation of the UAV trajectory that ensures it is at all times close to the target surface. But further, the dense point normals are used to determine spray gun orientation, to ensure that paint is sprayed in a direction normal to the surface. This generates a uniform painting result even on a surface with convexities



(a) User provided pattern (b) Painting result when 3D model is not used (c) Result from our approach that uses the 3D model

Fig. 10: Improvement in painting quality by using 3D model.

and concavities. In a scenario where the dense 3D model is not available, one can imagine performing an experiment where the UAV follows the provided painting pattern in a 2D plane close to the target surface with the spray gun always pointing forward. This particular experiment is chosen as a baseline to compare our approach. Fig. 10 shows a qualitative comparison of the painting result obtained from the baseline experiment versus our approach for a given pattern. The latter performs significantly better, justifying the approach and the need for a dense 3D model of the target surface.

D. Qualitative Results

Fig. 11 shows examples of area filling and line painting in thermal imagery. Fig. 11a, 11d are painted on a 3D rock surface, whereas, Fig. 11b, 11c are painted on a flat surface. This shows the adaptability to different target surfaces.

VIII. Conclusion

This paper has described the PaintCopter, an autonomous UAV for spray painting. The motivation is to do accurate themed/styled painting on 3D surfaces, while avoiding the need for scaffolding and ladders. We demonstrated the ability to do area fill painting in order to lay down a base color on a 3D surface, and line painting to overlay striations. A key future challenge is to paint color gradients. In summary, this paper has defined a general formulation for the robotic spray painting problem, has described challenges and proposed solutions, and hopefully has provided valuable insights into the problem by showing a full pipeline and experimental results.

References

[1] S.-H. Suh, I.-K. Woo, and S.-K. Noh, "Development of an automatic trajectory planning system (atps) for spray painting robots," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*. IEEE, 1991, pp. 1948–1955.

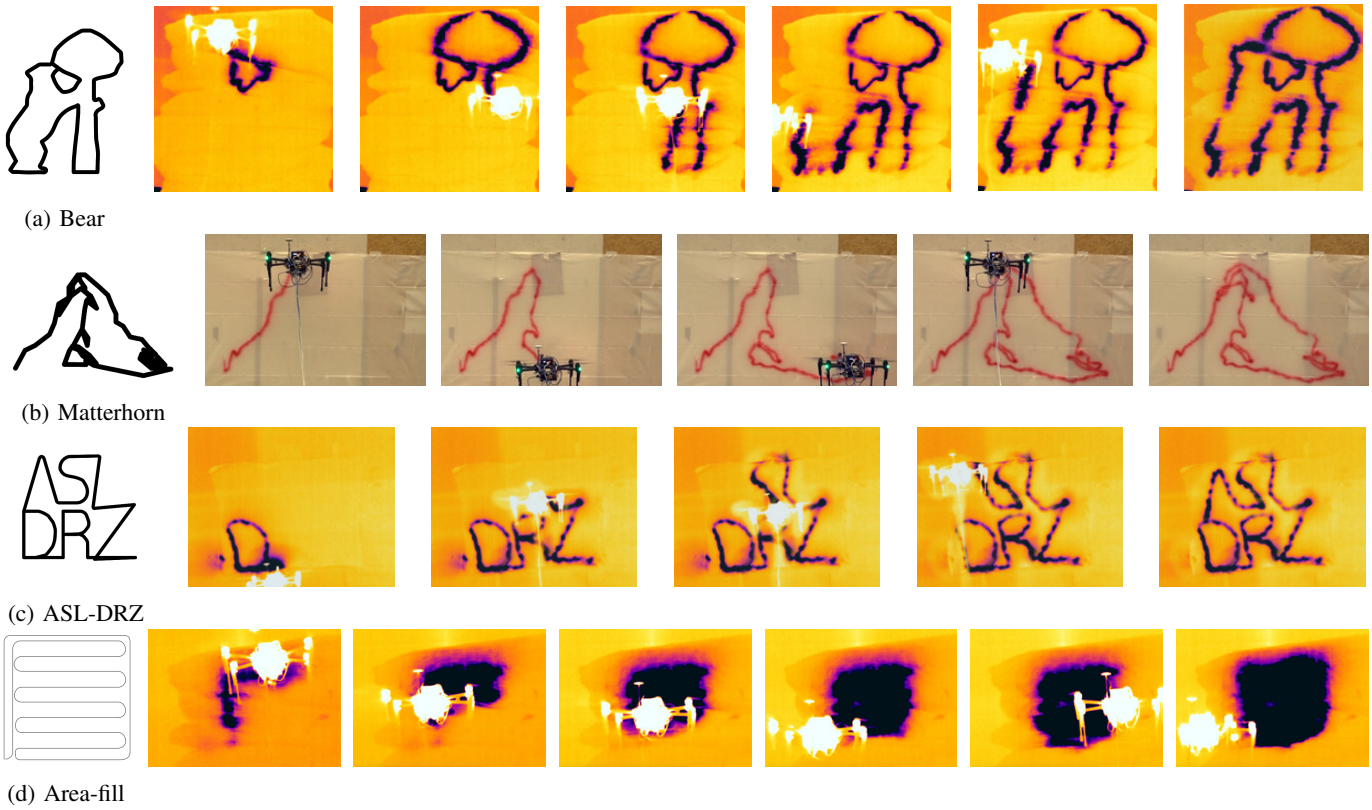
[2] M. V. Andulkar, S. S. Chiddarwar, and A. S. Marathe, "Novel integrated offline trajectory generation approach for robot assisted spray painting operation," *Journal of Manufacturing Systems*, vol. 37, pp. 201–216, 2015.

[3] A. Cortellessaa, S. Seriania, P. Gallinaa, S. Carratoa, M. Sortinob, S. Belfiob, and G. Ramponia, "Automatic path-planning algorithm for realistic decorative robotic painting," in *3rd International Conference, Production Engineering and Management, PEM, Trieste*, vol. 55, 2013.

[4] R. Prévost, A. Jacobson, W. Jarosz, and O. Sorkine-Hornung, "Large-scale painting of photographs by interactive optimization," *Comput. Graph.*, vol. 55, no. C, pp. 108–117, Apr. 2016. [Online]. Available: https://doi.org/10.1016/j.cag.2015.11.001

2377-3766 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2846278, IEEE Robotics and Automation Letters



(a) Bear

(b) Matterhorn

(c) ASL-DRZ

(d) Area-fill

Fig. 11: Area fill and line painting results. *Note:* For longer duration missions **(a, c)**, images at earlier and later timesteps have been overlaid to show the full design, because the water evaporates quickly from the surface and its thermal signature disappears. **(a-c)** use motion capture system for UAV tracking whereas **(d)** uses SLAM. **(b)** uses paint and the rest use water.

[5] R. Shilkrot, P. Maes, J. A. Paradiso, and A. Zoran, "Augmented airbrush for computer aided painting (cap)," *ACM Trans. Graph.*, vol. 34, no. 2, pp. 19:1–19:11, Mar. 2015. [Online]. Available: http://doi.acm.org/10.1145/2699649

[6] B. S. Faiçal, H. Freitas, P. H. Gomes, L. Y. Mano, G. Pessin, A. C. de Carvalho, B. Krishnamachari, and J. Ueyama, "An adaptive approach for uav-based pesticide spraying in dynamic environments," *Computers and Electronics in Agriculture*, vol. 138, pp. 210–223, 2017.

[7] B. Galea, E. Kia, N. Aird, and P. G. Kry, "Stippling with aerial robots," in *Proceedings of the Joint Symposium on Computational Aesthetics and Sketch Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering*. Eurographics Association, 2016, pp. 125–134.

[8] B. Galea and P. G. Kry, "Tethered flight control of a small quadrotor robot for stippling," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 1713–1718.

[9] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013, pp. 1280–1286.

[10] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct ekf-based approach," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2015, pp. 298–304.

[11] S. Lynen, M. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to mav navigation," in *Proc. of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2013.

[12] M. Kamel, T. Stastny, K. Alexis, and R. Siegwart, "Model predictive control for trajectory tracking of unmanned aerial vehicles using robot operating system," in *Robot Operating System (ROS) The Complete Reference, Volume 2*, A. Koubaa, Ed. Springer.

[13] A. S. Vempati, I. Gilitschenski, J. Nieto, P. Beardsley, and R. Siegwart, "Onboard real-time dense reconstruction of large-scale environments for uav," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 3479–3486.

[14] I. Sa, M. Kamel, M. Burri, M. Bloesch, R. Khanna, M. Popovic, J. Nieto, and R. Siegwart, "Build your own visual-inertial drone: A cost-effective and open-source autonomous drone," *IEEE Robotics Automation Magazine*, vol. PP, no. 99, pp. 1–1, 2017.

[15] I. Sa, M. Kamel, R. Khanna, M. Popović, J. Nieto, and R. Siegwart, "Dynamic system identification, and control for a cost-effective and open-source multi-rotor mav," in *Field and Service Robotics*. Springer, 2018, pp. 605–620.

[16] W. S. Cleveland and S. J. Devlin, "Locally weighted regression: an approach to regression analysis by local fitting," *Journal of the American statistical association*, vol. 83, no. 403, pp. 596–610, 1988.

[17] Q.-Y. Zhou, J. Park, and V. Koltun, "Fast global registration," in *European Conference on Computer Vision*. Springer, 2016, pp. 766–782.

[18] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics Research*. Springer, 2016, pp. 649–666.

[19] M. Burri, H. Oleynikova, , M. W. Achtelik, and R. Siegwart, "Real-time visual-inertial mapping, re-localization and planning onboard mavs in unknown environments," in *Intelligent Robots and Systems (IROS 2015), 2015 IEEE/RSJ International Conference on*, Sept 2015.

[20] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, *Robot Operating System (ROS): The Complete Reference (Volume 1)*. Cham: Springer International Publishing, 2016, ch. RotorS—A Modular Gazebo MAV Simulator Framework, pp. 595–625. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-26054-9_23

[21] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2011, pp. 3400–3407.

2377-3766 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.