IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), Würzburg, Germany, September 2019.

Reconstruction of Textured Meshes for Fire and Heat Source Detection

Radu Alexandru Rosu, Jan Quenzel, and Sven Behnke

Abstract— **Automated fire and heat source detection is a helpful and important capability of a robotic system to assist firefighters in various search and rescue (SAR) scenarios. In this paper, we investigate thermal mapping using textured meshes from preregistered LIDAR scans and show how to detect and localize heat sources therein. Further, we propose a novel occupancy mapping approach based on a sparse permutohedral lattice with a contradiction indicator deduced from ray-tracing the mesh. We evaluate our system on three datasets recorded with a micro aerial vehicle (MAV) including heat sources in a hall and real flames at a fire brigades training site. Dynamic objects are removed from the mesh, heat sources are located and their sizes are estimated.**

I. Introduction

Recent advances in robotic systems promise new solutions to support emergency forces in Search and Rescue (SAR) scenarios [1]. When firefighters approach a disaster site, it is mandatory to obtain an overview quickly and to continuously monitor the situation [2], especially if lives are at stake. Robotic systems exhibit a large potential to aid in this undertaking while reducing the risk for emergency personnel [3]. The firefighters in the recent *Notre-Dame de Paris* cathedral fire had to withdraw from the cathedral nave after the spire was in imminent danger of collapsing. Instead, a remote-controlled unmanned ground vehicle (UGV) was deployed for continued fire extinguishing and monitoring [4].

While such UGVs are nowadays a rare occurrence in fire brigades, micro aerial vehicles (MAVs) are increasingly used in recent years. MAVs equipped with thermal and color cameras have proven themselves as versatile tools delivering unobstructed views for points of interest. The real-time footage enables mission specialists to see the extent of the disaster. Close-up views, e.g. of collapsed [5] or burning buildings, are readily available without endangering the disaster response teams [6]. Meanwhile, thermal sensors allow to perceive people and to partially see through smoke and detect the seat of fire. Thus, an MAV can help to optimally position the fire hose without direct line of sight to the fire.

Nowadays, most disaster response UGVs and UAVs are remote-controlled [7] and as such limited by the sensors field-of-view, the latency for control input, radio range and distance to the operator. Insufficient sensor coverage or loss of communications, even if temporary, can cause the loss of the robot and potentially inflict further damage, e.g. from a crashing MAV. Hence, the A-DRZ project[1] aims to promote the development of ready-to-use autonomous rescue robotics.

The main contribution of this paper is a novel reconstruction system for textured meshes (Fig. 1) with automated localization of heat sources. We mesh preregistered LIDAR scans and fuse thermal as well as color images in textures. The textured mesh allows us to show high visual detail with simple geometric structure. Meshed dynamic obstacles tend to create tube-like structures that reduce the overall texture accuracy. We tackle these artifacts with ray-tracing through the mesh and determine occupancy with a sparse permutohedral lattice. The textures, fused from thermal as well as color images, provide a meaningful visualization for disaster response teams enriched with an automated detection and localization of heat sources. We demonstrate our thermal mapping approach on two different datasets including multiple real fires at a firefighting training site. An accompanying video is available[2].

II. Related Work

Thermal mapping [8] for building inspection typically relies on simultaneously captured color and thermal images from different poses and employs standard photogrammetry pipelines. A Structure-from-Motion (SfM) system registers color images in a common reference frame and generates a sparse 3D point cloud. Multi-View-Stereo (MVS) then generates a denser point cloud which is meshed and converted into simple Building Information Models (BIM). Thermal images are co-registered to RGB images e.g. using ground control points (GCP) and overlayed as a texture [9].

In robotic applications, a sparser colorized point cloud from few images (< 100) is often sufficient since computational resources and time-constraints are more restrictive. For phenotyping applications, Shafiekhani et al. [10] associate thermal values via projection to the SfM/MVS generated point cloud of calibrated stereo color cameras.

Sentenac et al. [11] use as well an SfM approach to triangulate matched points from stereo thermal cameras which are placed on the end effector of a Cartesian robot to perform thermomechanical analysis.

Truong et al. [12] generate two separate point clouds from color and thermal images by SfM and MVS. Since each SfM reconstruction is only upto scale, the correct metric scaling is computed with the calibrated relative transformation. Finally, iterative closest points (ICP) further improves the registration and allows to superimpose thermal and colored point clouds.

This work has been supported by the German Federal Ministry of Education and Research (BMBF) in the project "Kompetenzzentrum: Aufbau des Deutschen Rettungsrobotik-Zentrums" (A-DRZ) and as part of the research group FOR 1505 of German Research Foundation (DFG).

All authors are with the Autonomous Intelligent Systems Group, University of Bonn, Germany rosu@ais.uni-bonn.de

[1]https://rettungsrobotik.de
[2]https://www.ais.uni-bonn.de/videos/SSRR_2019_mesh

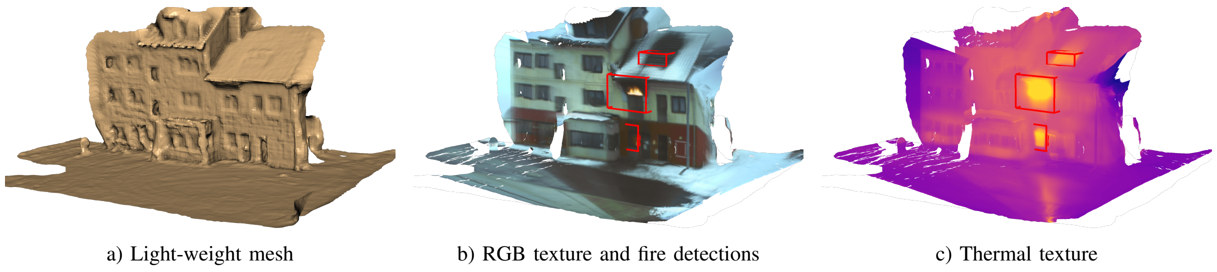


Fig. 1: Textured meshes: We reconstruct a light-weight mesh of the scene geometry using Poisson reconstruction (left). RGB and thermal data is fused into high resolution textures. The globally fused thermal texture (right) is used to detect heat sources and estimate their extent (middle).

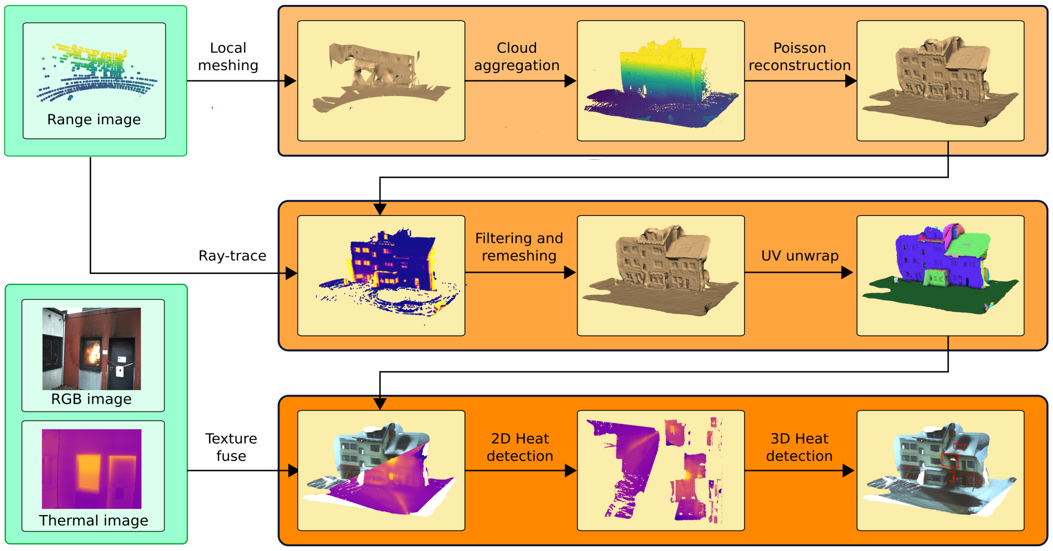


Fig. 2: Pipeline: Individual range images are used to create local meshes for fast normal estimation and cloud simplification. The resulting points, equipped with normals, are aggregated into a global point cloud. Poisson reconstruction is employed to extract a mesh. In a second step, the range images are used anew to ray-trace through the created mesh and map the occupancy using a permutohedral lattice. Points belonging to moving objects are discarded and the resulting cloud is remeshed. Finally, the mesh is simplified and unwrapped to provide a light-weight scene representation. In the third step, individual RGB and thermal images are fused onto the mesh. The thermal texture is binarized and used to detect heat sources on the 2D plane. The detections are lifted to 3D where they are further merged to yield the final heat source detections.

Yamaguchi et al. [13] superimpose thermal images on the estimated 3D point cloud of a Visual Odometry (VO) algorithm generated from color images. MVS is run separately on both modalities. A small triangulated mesh allows to overlay the current thermal image in the corresponding color image.

LIDAR and RGB-D sensors provide a vast amount of metric range measurements and allow for skipping the costly SfM and MVS steps. Instead, iterative closest point ICP aligns different scans. Voxel-based maps are predominantly used for reconstruction and meshed with marching cubes [14]. For instance, Vidas et al. [15] perform online mapping with an RGB-D and a thermal camera. Feature-based VO is fused with an ICP alignment. Keyframes are merged in a voxel grid and colored through ray-tracing along the view direction. Instead, Mittal et al. [16] aggregate depth images from an MAV in a voxel-based OctoMap, for path planning and landing site detection in rough terrain, as well as an RGB-vertex-colored mesh with VoxBlox for inspection by the ground personnel.

Schönauer et al. [17] aimed to provide firefighters with an augmentation based on an RGB-D and thermal cameras that overlays the live image on the ray-traced truncated signed distance function (TSDF) image from Kinect-Fusion. For the fusion of multiple thermal images, they store two frames per second, but did not provide more insights on the fusion itself. Marching cubes meshes the TSDF in a post-processing step. Their approach is only suitable for room-scale environments due to high memory requirements of the TSDF.

Borrmann et al. [18] combine a terrestrial LIDAR with a thermal and color camera on a UGV. A light-bulb grid helps to improve the calibration between LIDAR and thermal camera. Like us, they assume simultaneous image and scan acquisition, as well as preregistered scans from a simultaneous localization and mapping (SLAM) system. Points are projected to color and thermal images, then clustered per pixel and heuristically associated per cluster. The result is a colorized point cloud which is visualized with intensity, color or thermal information and allows to detect walls and windows in an automated fashion [19]. Their system captures only 9 images per scanner revolution with little overlap and only one revolution per pose, whereas we intentionally use image streams with high overlap and fuse multiple observations in order to obtain consistent textures. Similarly, Kim et al. [20] rotate three 2D LIDARs, a DSLR, and a thermal imager on a UGV to stitch panoramic images and point clouds.

A mapping based approach is taken by Zeise and Wagner [21]. They project LIDAR measurements into thermal images and aggregate the resulting point cloud in an OctoMap. The map is utilized to classify dielectric and metallic materials according to their emissivity based on the its viewing angle dependency.

Fritsche et al. [22] perform heat source detection directly on a fused 2D LIDAR and Mechanical Pivoting Radar point cloud. They process the scan further to estimate the robots pose. An occupancy grid map aggregates the scans and removes dynamic objects and spurious measurements. The detection clusters high-temperature points based on their Euclidean distance and stores the location, mean and temperature variance as well as the number of points in the cluster. In contrast, we acquire a 3D position and the approximate extent of the heat source while generating a 3D viewable representation on a mesh.

III. Overview

We present a novel approach to thermal map generation by coupling environment geometry with fused thermal information using textured meshes at independent resolutions. The independent resolution between the geometry and the texture allows to represent large scenes with a coarse mesh while the texture can be highly detailed. Our method operates in three steps: mesh generation, texturing and heat source detection.

In the *mesh generation* step [23], we first aggregate point clouds from range sensors assuming that an off-the-shelf SLAM system preregistered the scans into a common reference frame. We utilize a fast normal estimation based on an edge-maintaining local mesh with local line simplification. This heavily reduces the number of points aggregated without loss of detail while reducing time and memory requirements for global meshing. Poisson reconstruction [24] creates a global mesh from the aggregated point cloud with normals. QSlim [25] further simplifies the mesh. In a second run, we remove measurements of dynamic objects via ray-tracing through the mesh and filter occupancy with a sparse permutohedral lattice [26] before remeshing.

In the *texturing* step, we parameterize the mesh for texturing by adding seams and cuts in order to deform the mesh into the 2D plane. The thermal images, as well as RGB, are fused into individual textures.

In the *fire detection* step, we threshold on the thermal texture and find the extents of the fire.

Hence, the contribution presented in this article is fourfold:

- a scalable system for mesh creation from range measurements with coupled geometry and fused thermal data at independent resolution,
- a mesh-based dynamic object removal by means of ray-tracing,
- a fusion method for thermal images into a texture,
- thermal-based heat source detection on the texture.

IV. Notation

In the following, bold uppercase (lowercase) letters denote matrices (column vectors). The 4×4 matrix $\mathbf{T}_{F_2F_1}$ is a rigid transformation which maps points in homogeneous coordinates from coordinate frame F_1 to coordinate frame F_2. A subscript representing the frame is added when necessary, e.g. a point in world coordinates: $\mathbf{p}_w$. The pose $\mathbf{T}_F$ and the camera matrix $\mathbf{K}_F \in \mathbb{R}^{3\times3}$ project the point $\mathbf{p}_w$ into frame F. We assume the camera matrix to be derived from the standard pinhole model with focal length f_x, f_y, and principal point c_x, c_y. The projection of $\mathbf{p}_w$ into image coordinates $\mathbf{u} = (u_x, u_y)_F^\intercal \in \mathbf{\Omega} \subset \mathbb{R}^2$ is given by the following mapping:

$$g_F(\mathbf{p}_w) : \mathbf{p}_w \to \mathbf{p}_F, \tag{1}$$

$$(\mathbf{p}_F, 1)^\intercal = \mathbf{T}_{F_w} \cdot (\mathbf{p}_w, 1)^\intercal, \tag{2}$$

$$\pi_F(\mathbf{p}_F) : \mathbf{p}_F \to \mathbf{u}_F, \tag{3}$$

$$(x, y, z)_F^\intercal = \mathbf{K}_F \cdot \mathbf{p}_F, \tag{4}$$

$$\mathbf{u}_F = (x/z, y/z)^\intercal. \tag{5}$$

$I(\mathbf{u}) : \mathbf{\Omega} \to \mathbb{R}^n$ denotes an image or a texture, where $\mathbf{\Omega} \subset \mathbb{R}^2$ maps from pixel coordinates $\mathbf{u} = (u_x, u_y)^\intercal$ to n-channel values.

V. Methodology

Our system (Fig. 2) requires an input sequence of organized point clouds[3] $\mathcal{P}^t$, thermal and RGB images I^t (t indicates the time step). We assume preregistered point clouds in a common reference frame, and given extrinsic calibration $\mathbf{T}_{cd}$ from depth sensor to camera, as well as camera matrices. The depth sensor can be an RGB-D camera or a laser scanner. The output of our system is fourfold:

- a triangular mesh of the scene geometry, defined as a tuple $\mathcal{M} = (\mathcal{V}, \mathcal{F})$ of vertices $\mathcal{V}$ and faces $\mathcal{F}$. Each vertex $\in (\mathbb{R}^3 \times \mathbb{R}^2)$ contains a 3D point and a UV texture coordinate. A mesh face is represented by the indices $\in \mathbb{N}^3$ of the three spanning vertices within $\mathcal{V}$.
- a thermal texture S representing the heat radiated from the surface,
- a set of axis-aligned bounding boxes of heat sources,
- an RGB texture C representing the surface appearance.

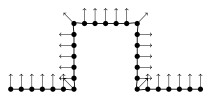
[3]An organized point cloud exhibits an image resembling structure, e.g. from commodity RGB-D sensors.

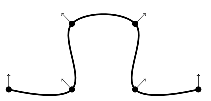
We first describe our depth preprocessing in Sec. V-A. Afterwards, we detail the mesh generation and parametrization (Sec. V-B), before elaborating on the thermal and color integration (Sec. V-C) and heat source detection (Sec. V-D).

A. Depth Preprocessing

We construct the global mesh out of a set of organized point clouds $\{\mathcal{P}^t\}$ obtained from a depth sensor. Surface reconstruction typically requires precise per-point normals. The naive way to estimate these is searching for the k-nearest-neighbors in the fully aggregated global point cloud. However, this requires a spatial subdivision structure, like a k-d tree, and easily grows to a considerable size for large point clouds, thus limiting the scalability. Instead, we exploit the inherent point cloud structure for fast normal estimation. Depth images from RGB-D sensors allow to directly query adjacent neighboring points. Rotating lidars generate organized scans. A complete revolution of e.g. a Velodyne VLP-16 produces a 2D array of size $16 \times N$ containing the measured range for each recorded point, where N is determined by the speed of revolution of the laser scanner.

Given this organized structure, we create a triangular local mesh with approximated normals as proposed by [27]. However, using all the points would introduce unnecessary large sets of redundant points that share common planes which would slow down the subsequent step of global meshing through Poisson reconstruction. Thus, we simplify each scan individually without sacrificing geometrical fidelity and reconstruct the mesh in a faster and more memory efficient manner (Fig. 5). For that, we employ a modified Ramer-Douglas-Peucker [28] line simplification which is applied on each scan ring. We introduce additional offset points around simplified edges to constrain the normals of points and allow the subsequent surface reconstruction to maintain hard edges and sharp features (Fig. 3).


(a) Original scanline


(b) Simplified scanline

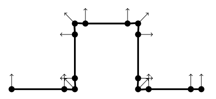

(c) Simplified with offset points

Fig. 3: Line simplification: The original scan line (left) is excessively dense in planar areas. The original simplification greatly reduces the number of points but creating a global surface using a method like Poisson reconstruction overly smoothes the edges (middle). We add further constraints which allow Poisson reconstruction to maintain sharp features (right).

We create the local mesh in 2D by unwrapping the scan using polar coordinates and perform a constrained 2D Delaunay triangulation. Simplified segments from the line simplification are added as constrained edges in the triangulation. This ensures that points that lie on the same scan ring will be connected together by triangles. The resulting triangulation is not optimal when lifted back to 3D as it

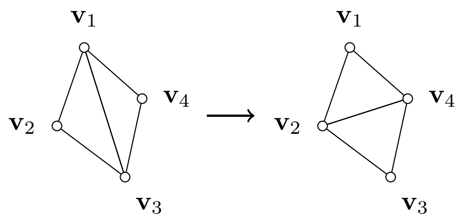


Fig. 4: Edge flipping: Badly conditioned triangle pairs have their edge flipped to promote more equilateral triangles. This allows the vertices to connect to closely neighboring ones and improves the subsequent step of normal estimation.

does not take the anisotropic sampling of the laser scanner into account. Hence, we perform an iterative local mesh refinement via edge flipping that prefers equilateral over acute-angled triangles (Fig. 4). This ensures that vertices will be connected to spatially closer vertices. Afterwards, we use two edges per face to calculate the face normal from the cross product. Per vertex normals are then estimated via a *Mean Weighted by Angle* (MWA) scheme [29] from adjacent faces.

B. Mesh Generation

We employ Poisson reconstruction to recover a high quality mesh after aggregating the points and normals from the simplified scans. Since in geometrically simple areas like the ground meshes are often overly dense, we apply a second global simplification step with QSlim [25]. So far, dynamic objects are still represented in the mesh and introduce artifacts where there should be free space. Hence, we identify erroneous triangles by ray-tracing all points through the mesh. Per simplified scan this runs in real-time on the CPU with the Embree library [30]. We decided to use a mesh rather than a voxel grid for efficiency reasons. In general, fewer triangles need to be checked for intersection during ray-tracing than voxels while triangles better represent the underlying geometry especially for flat surfaces common in man-made environments and are more adaptive in size to the surface.

We use an indicator function δ to define contradiction between the ray and the triangle:

$$\delta = \begin{cases} 0 & \text{if } w_{view} \cdot (d_x - d_i) \leq \gamma \\ 1 & \text{otherwise} \end{cases}, \tag{6}$$

$$w_{view} = (\mathbf{o}_w - \mathbf{p}_{w,i}) \cdot \mathbf{n}_x. \tag{7}$$

Here, d_i is the distance along the viewing ray of the laser beam to the point of intersection. Obviously, the distance d_x from sensor to laser beam end point is always larger or equal to d_i. The weighting factor w_{view} additionally downweighs steep incident angles using the dot product between viewing ray from camera origin $\mathbf{o}_w$ to the intersection point $\mathbf{p}_{w,i}$ and surface normal $\mathbf{n}_x$. Hence, if the weighted distance is too large ($> \gamma$), the ray contradicts the triangle.

In order to remove the points that correspond to dynamic object we create an occupancy map in which we store the

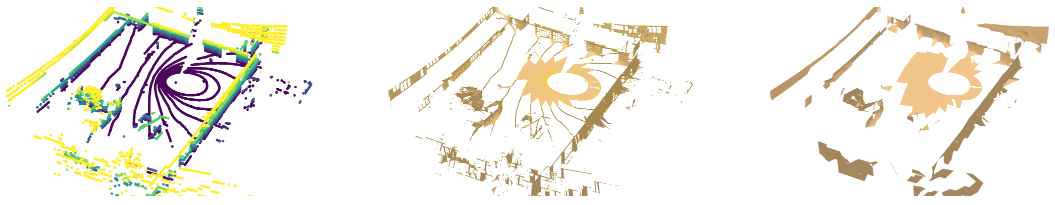


Fig. 5: Depth preprocessing: During sudden movements of the laser scanner, the scan rings are compressed behind and expanded in front of the sensor (left). This creates many small and steep triangles which degrades normal estimation. We perform iterative edge-flipping in order to connect each vertex with their closest neighbor, hence, improving the likelihood for estimating correct normals. Furthermore, we apply line simplification to each scan ring independently for data reduction without sacrificing mesh fidelity.

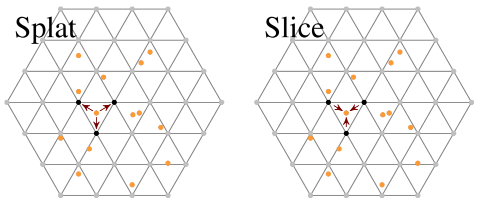


Fig. 6: Splatting and slicing on the Permutohedral Lattice: Splatting distributes the values of each point onto the corners of its corresponding simplex according to the barycentric coordinates. Slicing is the inverse operation and computes the output value of a point by sampling the corners of the simplex using barycentric interpolation. Figure adapted from Kiefel et al. [31].

indicator function δ. Traditional occupancy mapping employs grids to store the probability of a cell being occupied. Instead, we use a sparse permutohedral lattice [26] comprised of regularly ordered d-dimensional simplices with $d + 1$ vertices each. In our case d is 3 since we work with data in three-dimensional space. This space representation has the advantage of scaling linearly with increasing dimensionality compared to grids w.r.t. memory consumption and number of vertices. To store the indicator function, we use a splatting operation which consists of distributing the values of each point onto the corners of the corresponding simplex (Fig. 6). This creates an occupancy map which is piece-wise linear as opposed to the piece-wise constant of traditional grids. Slicing is the inverse operation of splatting and it computes the value of a point in space by interpolating between the values stored in the vertices of the simplex.

For each intersected triangle, we check the enclosing simplex of the intersecting point, allocate it if needed and splat the indicator function in homogeneous coordinates. The homogeneous part serves as a counter which results in a weighted running average. Once all points are ray-traced and all indicator functions are embedded in the lattice, we re-aggregate the point cloud for meshing and determine which points should be discarded. For this, we obtain for each scan point a carving coefficient from the enclosing simplex through slicing. We threshold on the carving coefficient and disregard points with a high coefficient—thus removing dynamic points. The static point cloud is again meshed and simplified.

We finally parameterize the global mesh for texturing to obtain UV coordinates per vertex (Fig. 8). For that, we make use of the *UV smart project* function provided within Blender [32].

C. Thermal and Color Integration

In this section we detail our approach on how to update the global 16 bit thermal texture S using individual images. In addition, we also fuse the raw color images I^t into a global 3-channel color texture C. We first perform a visibility check, inspired by *shadow mapping techniques* in computer graphics, prior to updating the global textures. Hence, we calculate per texel x with UV coordinates $\mathbf{u}_x$ its 3D point $\mathbf{p}_x$ via barycentric interpolation from the face vertices. We then render for the current camera pose the depth map D and project each texel point $\mathbf{p}_x$ into the view. If the distance towards the texel (d_x) is larger than the value within $D(\pi(g_F(\mathbf{p}_x)))$, the texel lies behind the visible part and will be discarded. We additionally use a small ϵ set to 1×10^{-3} to account for possible numerical and discretization issues when rendering the depth map. We thus use a per texel indicator variable $r_x \in \{0, 1\}$ to indicate occlusion:

$$r_{x_i} = \begin{cases} 1, & \text{if } d_x \geq D(\pi_F(g_F(\mathbf{p}_x))) + \epsilon \\ 0, & \text{otherwise} \end{cases} . \tag{8}$$

All remaining texels $(r_x < 1)$ are fused with a weighted running average:

$$C(\mathbf{u}_x)^t = \frac{W(\mathbf{u}_x)^{t-1} C(\mathbf{u}_x)^{t-1} + w_x I^t(\pi_F(g_F(\mathbf{p}_x)))}{W(\mathbf{u}_x)^{t-1} + w_x}, \tag{9}$$

$$W(\mathbf{u}_x)^t = W(\mathbf{u}_x)^{t-1} + w_x. \tag{10}$$

For the case of thermal images we fuse into the global thermal texture S and I^t then denotes the current thermal image at time t. The weight w_x takes the distance, the radial

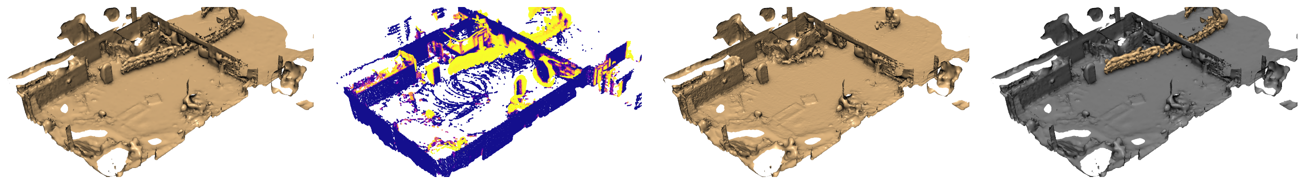


Fig. 7: Dynamic object removal: An initial mesh is created by using the point cloud from a laser scanner. The point cloud is then ray-traced into the mesh and an occupancy map is stored in the vertices of a permutohedral lattice (middle). The points falling in areas of high contradiction with the mesh (yellow) are discarded and the resulting point cloud is remeshed (right).

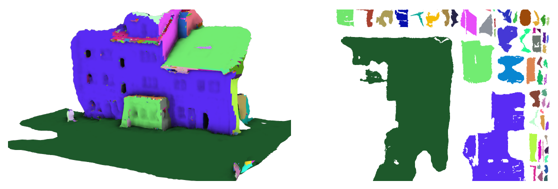


Fig. 8: UV parameterization: Roughly planar regions are clustered together and greedly projected onto the 2D plane to obtain UV coordinates for each vertex of the mesh.

intensity fall-off, and the viewing angle into account:

$$w_x = w_{dist} \cdot w_{vign} \cdot w_{thermal}, \tag{11}$$

$$w_{dist} = (\|g_F(\mathbf{p}_{w,x})\|_2^2 + \epsilon)^{-1}, \tag{12}$$

$$w_{vign} = \cos(\theta_x)^4, \tag{13}$$

$$w_{thermal} = w_{view} \cdot \left(1 - (\sigma\sqrt{2\pi})^{-1} \mathrm{e}^{-\frac{1}{2}\left(\frac{\tau-\mu}{\sigma}\right)^2}\right), \tag{14}$$

$$\tau = \arccos\left(w_{view}\right). \tag{15}$$

Here, w_{dist} is the inverse distance from the texel to the camera, which promotes frames that are spatially closer to the mesh, improving the resolution and accuracy. A small ϵ value prevents division by zero. The angle θ_x between reprojection of the texel and the principal axis of the camera is used to account for the radial decrease in intensity following the $\cos^4$ law [33].

The viewing angle w.r.t. the surface normal influences the measurement accuracy of a thermal camera, especially for surfaces with high reflectivity and low emissivity [34]. Hence, we model the reliability of the thermal camera by multiplying w_{view} (cosine of the viewing angle) with one minus a Gaussian. The Gaussian reduces the influence of direct reflection at close to right angles. Fig. 9a shows the resulting weighting term $w_{thermal}$ w.r.t. the viewing angle.

For color fusion we replace the third term with w_{view} from Eq. 7. The weight increases for texels imaged from a frontal perspective with the camera origin at $\mathbf{o}_w$.

Visualizing the full 16 bit range of the thermal texture with on-screen 8 bit resolution requires an appropriate scaling such that details will be visible. This is especially important in scenes with strong variations e.g. regions of fire in a normal tempered environment. A linear scaling would create a mostly uniform image where only the fiery sections

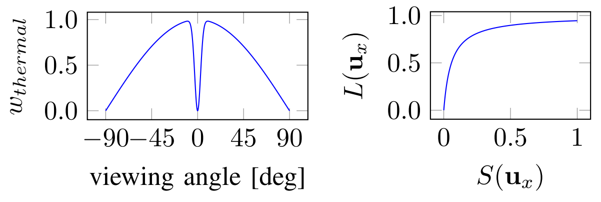


Fig. 9: Thermal weighting function w.r.t. viewing angle and tone mapping operator which assigns more range to low values.

are distinguishable. Hence, we use a simple tone mapping operator [35]:

$$\overline{L} = \left(|\Omega|^{-1} \sum_{\mathbf{u}\in\Omega} S\left(\mathbf{u}\right)\right)^{-1}, \tag{16}$$

$$L\left(\mathbf{u}_x\right) = \frac{S\left(\mathbf{u}_x\right) \cdot \overline{L}}{1 + S\left(\mathbf{u}_x\right) \cdot \overline{L}}. \tag{17}$$

This non-linear mapping squishes higher intensities while stretching lower intensities as visualized in Fig. 9b. Computing the tone-mapped image is efficiently computed on the GPU by first calculating $\overline{L}$ as the value of the 1×1 coarsest image in the pyramid of S, followed by a pixelwise-operation.

D. Heat Source Detection

The detection of heat sources is performed by first binarizing the 16 bit thermal texture S on GPU. The binarized image passes through the contour detection of OpenCV [36] which outputs a list of points along the contours of the heat source together with the grouping of the points into continous segments delimiting the borders of a heat source in texture space. The seams and cuts added by the UV parameterization cause gaps in texture space even though texels may be close in the actual mesh. In order to merge the segments in 3D we calculate axis-aligned bounding boxes (AABB) for each segment from the 3D position $\mathbf{p}_x$ of the contour texels and greedily merge intersecting bounding boxes to obtain the final detections. For visualization purposes, we only draw the edges of the remaining AABBs as shown in Fig. 10.

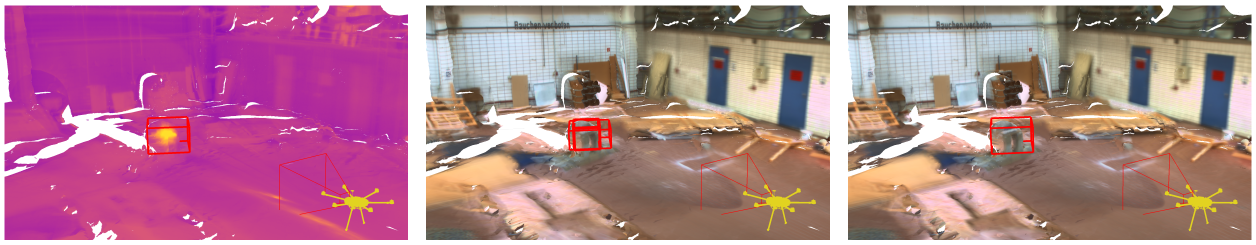

a) Thermal map with detections b) Detections c) Merged detections

Fig. 10: Merging of detected heat sources: Detections of heat sources from the thermal texture (left) are lifted to 3D yielding multiple overlapping regions (middle). The estimated bounding boxes are greedly merged to yield a final detection (right).

Stage	Step	Time
Meshing	Local mesh	58.3ms
	Aggregation	1.2ms
	Poisson[†]	6.6s
	QSlim[†]	5.7s
	Unwrapping[†]	26s
Dynamics	Raytrace	67.5ms
	Splat	32.3ms
	Slice	9.6ms
Texture	Texture fuse	14.8ms
	Fire detect	12.6ms

TABLE I: The run-time for the different parts of our pipeline. The timing for each step is the average run-time to process one laser scan. The steps with [†] indicate batch processes which run for a full aggregated point cloud or a full mesh.

Cloud	#Points	#Verts	Time(s)	Mem(MB)
naïvely	6.5M	0.15M	10.2	162
simplified	0.8M	0.13M	6.6	151

TABLE II: Poisson reconstruction using the naïvely aggregated cloud and our edge-aware simplified cloud. We report the number of points of the input cloud, the number of vertices of the reconstructed mesh, and the time and peak memory used by the reconstruction process.

VI. Evaluation

We perform all experiments on a Laptop equipped with an Intel Core i7-8550U CPU, 16 GB RAM and a dedicated Nvidia MX150 mobile GPU with 2 GB VRAM.

For the first two experiments, we flew with a DJI Matrice 600 MAV equipped with a stereo rig of Point Grey BlackFly-S U3-51S5C-C color cameras, a FLIR Boson 640 thermal camera and a Velodyne Puck LITE LIDAR. The heat source in the initial test was a brick on a stove in our hall. The pilot flew in from outside and followed the MAV through the door. Hence, the initial mesh (Fig. 7a) contains a tube-like structure which is in contradiction to many range measurements and vanishes after point removal and remeshing. Further contradictions stem from Poisson reconstruction artifacts and windows. The heat source is clearly visible in the thermal map (Fig. 10) and successfully detected (red box). The parts of the pipeline that are performed for each laser scan are capable of running in real time while batch processes like Poisson reconstruction, mesh simplification with QSlim and UV unwrapping run for several seconds. An analysis of the run-time of each component is provided in Tab. I. Furthermore we include an analysis of the run-time of Poisson reconstruction for a point cloud in which every point of the laser scan is naïvely aggregated and compare it with our method of 1D line simplification in Tab. II.

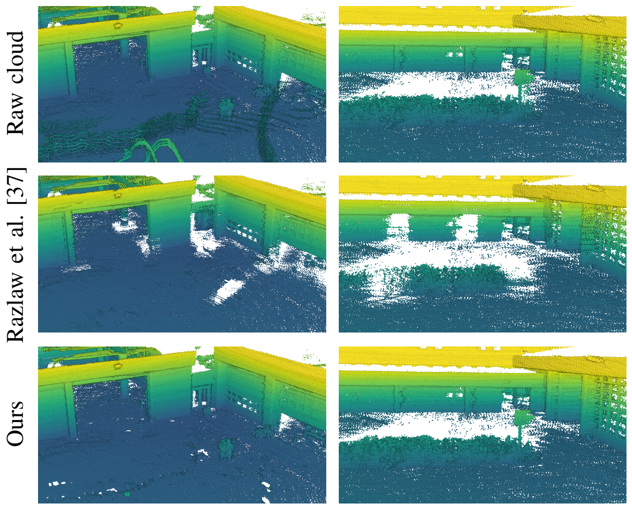


Fig. 11: Comparison on dynamic point removal against method by Razlaw et al. [37].

The second experiment took place on a snowy day at the firefighting training facility of the fire brigade Dortmund. There are three fires with increasing detection difficulty installed. The flame on the first floor above a window is directly visible. The second fire is at an angle behind a window on the ground floor and the third is below the roof. The last two fires are only visible in RGB images from a certain angle or may be inferred from the missing snow on the roof but not in summer. In contrast, our detection finds them since all three are clearly visible on the thermal texture in Fig. 1.

Our third experiment evaluates the capabilities of the proposed dynamic object removal in a more challenging setup. We compare our method against Razlaw et al. [37] on their LIDAR dataset recorded from the same MAV with four people running around. Fig. 11 shows the resulting point clouds. While both methods remove the traces, ours retains more details like pillars and a sign in the courtyard. Although, small regions of the ground plane are removed by the lattice, the meshing is likely to fill these up.

VII. Conclusion

In this paper, we proposed a novel thermal mapping system on textured meshes which couples thermal information with geometric representation at independent resolution while improving scalability and enforces spatial as well as temporal consistency over multiple observations. Ray-tracing through the mesh allows dynamic object removal via occupancy mapping with a sparse permutohedral lattice. We further showed how to utilize the texture for detection and localization of fires and heat sources. Although our system is currently a batch process, all individual steps except for the meshing run in real-time. Hence, we want to further investigate how to incorporate incremental meshing.

References

[1] F. Nex and F. Remondino, "UAV for 3D mapping applications: a review," *Applied Geomatics*, vol. 6, no. 1, pp. 1–15, 2014.

[2] C. Yuan, Y. Zhang, and Z. Liu, "A survey on technologies for automatic forest fire monitoring, detection, and fighting using unmanned aerial vehicles and remote sensing techniques," *Canadian Journal of Forest Research*, vol. 45, no. 7, pp. 783–792, 2015.

[3] D. Twidwell, C. R. Allen, C. Detweiler, J. Higgins, C. Laney, and S. Elbaum, "Smokey comes of age: unmanned aerial systems for fire management," *Frontiers in Ecology and the Environment*, vol. 14, no. 6, pp. 333–339, 2016.

[4] L. Peskoe-Yang. (2019) Paris firefighters used this remote-controlled robot to extinguish the notre dame blaze. [Online]. Available: https://spectrum.ieee.org/automaton/robotics/industrial-robots/colossus-the-firefighting-robot-that-helped-save-notre-dame

[5] I. Kruijff-Korbayová, L. Freda, M. Gianni, V. Ntouskos, V. Hlaváč , V. Kubelka, E. Zimmermann, H. Surmann, K. Dulic, W. Rottner, and E. Gissi, "Deployment of ground and aerial robots in earthquake-struck Amatrice in Italy (brief report)," in *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, 2016.

[6] G. M. Kruijff, F. Pirri, M. Gianni, P. Papadakis, M. Pizzoli, A. Sinha, V. Tretyakov, T. Linder, E. Pianese, S. Corrao, F. Priori, S. Febrini, and S. Angeletti, "Rescue robots at earthquake-hit Mirandola, Italy: A field report," in *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, 2012.

[7] M. Leingartner, J. Maurer, A. Ferrein, and G. Steinbauer, "Evaluation of sensors and mapping approaches for disasters in tunnels," *Journal of Field Robotics (JFR)*, vol. 33, no. 8, pp. 1037–1057, 2016.

[8] Y. K. Cho, Y. Ham, and M. Golpavar-Fard, "3D as-is building energy modeling and diagnostics: A review of the state-of-the-art," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 184 – 195, 2015.

[9] M. Scaioni, E. Rosina, A. L'Erario, and L. Dìaz-Vilariño, "Integration of infrared thermography and photogrammetric surveying of built landscape," *International Arch. Photogramm. Remote Sens. Spatial Inf. Sci. (ISPRS)*, vol. 42, 2017.

[10] A. Shafiekhani1, F. B. Fritschi, and G. N. DeSouza, "A new 4D-RGB mapping technique for field-based high-throughput phenotyping," in *BMVC*, 2018.

[11] T. Sentenac, F. Bugarin, B. Ducarouge, and M. Devy, "Automated thermal 3D reconstruction based on a robot equipped with uncalibrated infrared stereovision cameras," *Advanced Engineering Informatics*, vol. 38, pp. 203 – 215, 2018.

[12] T. P. Truong, M. Yamaguchi, S. Mori, V. Nozick, and H. Saito, "Registration of RGB and thermal point clouds generated by structure from motion," in *Workshop Proceedings of the IEEE International Conference on Computer Vision (ICCV Workshops)*, 2017.

[13] M. Yamaguchi, T. P. Truong, S. Mori, V. Nozick, H. Saito, S. Yachida, and H. Sato, "Superimposing thermal-infrared data on 3D structure reconstructed by RGB visual odometry," *IEICE TRANSACTIONS on Information and Systems*, vol. 101, no. 5, pp. 1296–1307, 2018.

[14] M. Zollhöfer, P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb, "State of the art on 3D reconstruction with RGB-D cameras," *Computer Graphics Forum*, vol. 37, no. 2, pp. 625–652, 2018.

[15] S. Vidas, P. Moghadam, and S. Sridharan, "Real-time mobile 3D temperature mapping," *IEEE Sensors Journal*, vol. 15, no. 2, pp. 1145–1152, Feb 2015.

[16] M. Mittal, A. Valada, and W. Burgard, "Vision-based autonomous landing in catastrophe-struck environments," *CoRR*, vol. abs/1809.05700, 2018. [Online]. Available: http://arxiv.org/abs/1809.05700

[17] C. Schönauer, E. Vonach, G. Gerstweiler, and H. Kaufmann, "3D building reconstruction and thermal mapping in fire brigade operations," in *Proceedings of the 4th Augmented Human International Conference*. ACM, 2013, pp. 202–205.

[18] D. Borrmann, J. Elseberg, and A. Nüchter, "Thermal 3D mapping of building façades," in *Proc. of Int. Conf. on Intelligent Autonomous Systems (IAS)*. Springer, 2013, pp. 173–182.

[19] G. Demisse, D. Borrmann, and A. Nüchter, "Interpreting thermal 3D models of indoor environments for energy efficiency," *Journal of Intelligent & Robotic Systems (JINT)*, vol. 77, no. 1, pp. 55–72, 2015.

[20] P. Kim, J. Chen, and Y. K. Cho, "Robotic sensing and object recognition from thermal-mapped point clouds," *International Journal of Intelligent Robotics and Applications (IJIRA)*, vol. 1, no. 3, pp. 243–254, Sep 2017.

[21] B. Zeise and B. Wagner, "Tempered point clouds and octomaps: A step towards true 3D temperature measurement in unknown environments," in *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, 2017, pp. 88–95.

[22] P. Fritsche, B. Zeise, P. Hemme, and B. Wagner, "Fusion of radar, LiDAR and thermal information for hazard detection in low visibility environments," in *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, 2017, pp. 96–101.

[23] R. A. Rosu, J. Quenzel, and S. Behnke, "Semi-supervised semantic mapping through label propagation with semantic texture meshes," *International Journal of Computer Vision (IJCV)*, Jun 2019.

[24] M. Kazhdan and H. Hoppe, "Screened poisson surface reconstruction," *ACM Transactions on Graphics (ToG)*, vol. 32, no. 3, p. 29, 2013.

[25] M. Garland and P. S. Heckbert, "Simplifying surfaces with color and texture using quadric error metrics," in *Proceedings of the International Conference Visualization (VIS)*, 1998, pp. 263–269.

[26] A. Adams, J. Baek, and M. A. Davis, "Fast high-dimensional filtering using the permutohedral lattice," in *Computer Graphics Forum*, vol. 29, no. 2. Wiley Online Library, 2010, pp. 753–762.

[27] D. Holz and S. Behnke, "Registration of non-uniform density 3D laser scans for mapping with micro aerial vehicles," *Robotics and Autonomous Systems*, vol. 74, pp. 318–330, 2015.

[28] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 10, no. 2, pp. 112–122, 1973.

[29] G. Thürrner and C. A. Wüthrich, "Computing vertex normals from polygonal facets," *Journal of Graphics Tools*, vol. 3, no. 1, pp. 43–46, 1998.

[30] I. Wald, S. Woop, C. Benthin, G. S. Johnson, and M. Ernst, "Embree: a kernel framework for efficient CPU ray tracing," *ACM Transactions on Graphics*, vol. 33, no. 4, p. 143, 2014.

[31] M. Kiefel, V. Jampani, and V. G. Peter, "Sparse convolutional networks using the permutohedral lattice," *CoRR*, vol. 2, 2015.

[32] Blender Online Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Blender Institute, Amsterdam, 2018. [Online]. Available: http://www.blender.org

[33] D. Goldman and J. Chen, "Vignette and exposure calibration and compensation," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2005.

[34] FLIR Systems Inc., "5 factors influencing radiometric temperature measurements," FLIR Systems Inc., Tech. Rep., 2016. [Online]. Available: https://www.flir.com/discover/oem/cores/5-factors-influencing-radiometric-temperature-measurements/

[35] E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda, "Photographic tone reproduction for digital images," *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 267–276, 2002.

[36] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[37] J. Razlaw, J. Quenzel, and S. Behnke, "Detection and tracking of small objects in sparse 3D laser range data," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2018.