An Efficient Locally Reactive Controller for Safe Navigation in Visual Teach and Repeat Missions

Matias Mattamala, Nived Chebrolu, and Maurice Fallon

Abstract— To achieve successful field autonomy, mobile robots need to freely adapt to changes in their environment. Visual navigation systems such as Visual Teach and Repeat (VT&R) often assume the space around the reference trajectory is free, but if the environment is obstructed path tracking can fail or the robot could collide with a previously unseen obstacle. In this work, we present a locally reactive controller for a VT&R system that allows a robot to navigate safely despite physical changes to the environment. Our controller uses a local elevation map to compute vector representations and outputs twist commands for navigation at 10 Hz. They are combined in a Riemannian Motion Policies controller that requires < 2 ms to run on a CPU. We integrated our controller with a VT&R system onboard an ANYmal C robot and tested it in indoor cluttered spaces and a large-scale underground mine. We demonstrate that our locally reactive controller keeps the robot safe when physical occlusions or loss of visual tracking occur such as when walking close to walls, crossing doorways, or traversing narrow corridors.

I. INTRODUCTION

Safe navigation is a fundamental capability required to successfully deploy robots in natural and human-made environments. Forests, mines, and industrial facilities are challenging places due to clutter, narrow passages, or moving objects that effect the configuration space in which a robot operates. Dealing with such difficulties requires a sense of local awareness, in order to effectively adapt the robot's behavior to the task and state of the environment.

Visual Teach and Repeat (VT&R) systems are a practical approach to enable inspections or patrols of known places or to navigate from point-to-point [1], [2]. Instead of building a precise metric map, VT&R relies on a *topo-metric* representation built during a *teach* phase, which is then used during the *repeat* phase to execute point-to-point missions using only camera input to track the reference path. However, in practice, the visual appearance of real environments can change (due to the lighting conditions or because of onboard illumination [3]) as can the physical layout (such as when a pathway is blocked), risking the success of the mission.

In this paper, we present a locally reactive controller that works as a safety layer to help a VT&R system achieve robust operation after significant changes to the environment. It allows a robot to be aware of its surroundings using a local map built online. We argue that for VT&R missions,
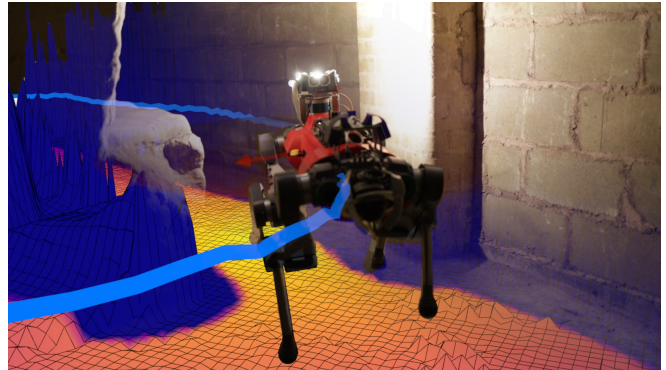
Fig. 1: Our reactive body controller allows an ANYmal robot to safely execute Visual Teach and Repeat navigation through narrow corridors in a decommissioned underground mine. We use the local map built on-the-fly to compute efficient vector field representations, such as the *geodesic distance field* pictured, and combine them in a Riemannian Motion Policies controller that runs at 10 Hz.

we do not need to rely on a global and a local planner to navigate these environments. Instead, we exploit the fact that the taught path effectively represents a *global plan in free space*, only requiring a fast, reactive controller to track the path safely, which is the main focus of this work.

Our controller uses the local map to generate expressive vector fields that provide the cues necessary for navigation, such as avoiding obstacles or reaching (local) goals around corners. The fields are combined in a principled manner using a Riemannian Motion Policy (RMP) formulation [4], which allows us to dynamically enable specific behaviors depending on the robot's state. We demonstrate its closed-loop integration with a VT&R system, showing safe and fast navigation in cluttered indoor spaces and challenging large scale environments.

The main contributions of our work are the following:

- A reactive RMP-based twist controller that uses a local elevation map built online as the main perceptual input.
- Integration of guidance cues for navigation using vector field representations which are computed from the local map at 10 Hz on a CPU.
- Closed-loop integration with our VT&R system [5] on a quadruped platform.
- Real-world validation using the ANYmal C quadruped in realistic environments including scenarios with substantial change to the scene and narrow pathways.
- First demonstration of a RMP reactive controller with a full-sized dynamic robot in large-scale environments.

II. RELATED WORK

A. Collision-free Navigation

The navigation problem aims to determine an action (or set of actions) that allows a robot to reach a specific goal without colliding with obstacles. This is typically achieved through *reactive controllers* or *local planners*. Reactive controllers leverage the information available at a specific time and determine a single action to be directly executed by the robot. In contrast, local planners typically generate a collision-free trajectory by considering a short look-ahead and solving an optimization problem over that region [7]. In this paper, we use a reactive controller based on RMP that functions as a safety layer on top of the low-level controller and can be integrated seamlessly with other local planners.

For legged robots, determining the optimal action or trajectory can involve the computation of footsteps, which is known as *whole-body control/planning*. Buchanan *et al.* [8] combine the whole-body planning problem with obstacle information in the form of a 3D signed distance field that is used for collision checking. This allows the robot to navigate through narrow spaces while planning individual footsteps through traversable regions. Similarly, Gaertner *et al.* [9] demonstrate collision-free locomotion by means of a Model Predictive Controller (MPC). These approaches couple the collision-free navigation task with footstep computation, thereby making them difficult to generalize to different platforms. Whereas, in this work, we follow a *pose* or *twist-based* approach that only considers the robot's torso or base link, making it directly usable for different ground robots. The twist-base control has been successfully demonstrated on several platforms such as ANYmal B [10], ANYmal C [11], Mini Cheetah [12], and Vision60 [13].

B. Representations for Local Navigation

Classical navigation approaches typically consider only the geometric information of the environment, which can be either represented by occupancy maps, voxels [14], point clouds [15], meshes [16], [17] or height maps [18]. Typically, a traversibility map is computed from these representations giving a measure of how safe a region is for the robot navigate. Semantics can often help to expand the notion of traversable space. Bradley *et al.* [19] learn four types of terrain used for navigation in a forest environment with the LS3 quadruped. Nardi and Stachniss [20] as well as Wellhausen *et al.* [21] learn terrain properties in a self-supervised fashion which are then used as costs for a path planning algorithm that enables them to navigate safely.

Representations such as Signed Distance Fields (SDFs) [22] are particularly suitable for navigation, since they encode the distance to the obstacle surfaces in the environment. The SDFs are also differentiable and lend themselves to computation of vector field gradients, which can be seamlessly integrated in optimization-based planners or vector-based controllers. Another useful representation is the Geodesic Distance Fields (GDF) which captures the shortest path to goal from from different positions in the map. These fields can be
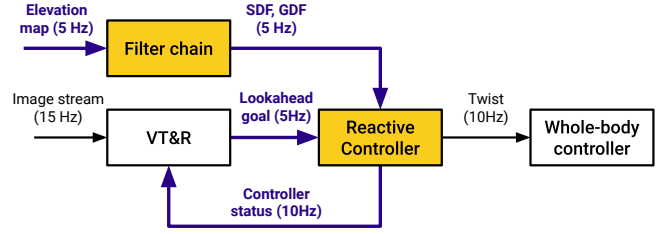

Fig. 2: Main pipeline of our safe VT&R system integrated with a quadruped robot. The *filter chain* module computes vector field representations from the elevation map (Sec. III-A). They are used by the *reactive controller* module (Sec. III-B) to generate the command sent to the whole-body controller.

computed efficiently using Fast Marching Method [23] or the Heat Method [24] and have been used in several approaches for navigation [25], [17], [26], [27]. In this work, we take advantage of these representations and use a combination of vector fields to generate a collision-free command.

C. Safe Navigation for VT&R

In the context of VT&R systems, only a few works have addressed safe navigation explicitly. Ostafew *et al.* [28] formulate the collion-free navigation problem as a non-linear MPC controller with constraint satisfaction. The constraints are given by potential obstacles or localization limits, i.e, forcing the robot to stay close to the reference path to avoid localization failures. Berczi and Barfoot [29] learn models of traversability over multiple runs, which are used to detect unexpected obstacles when compared to the reference experience. In our work, we exploit the traversability information gathered from multiple vector fields in local neighbourhood to ensure safe navigation.

Recently, Bista *et al.* [30] presented a topological navigation system for indoor environments which combined the output of a visual servoing system used for navigation with the solution of a local plan computed in a 2D grid with A*. Since we focus both on indoor and outdoor environments, our controller takes into account the 3D geometry of the environment both for traversability computation as well as cues used to compute the control command.

III. METHOD

Our locally reactive controller and its integration with the VT&R system is summarized in Fig. 2. Our approach is a safety mechanism operating on the robot's local elevation map. First, we perform a filtering step (*Filter chain* module, Sec. III-A) to determine the traversability of the local environment and to compute efficient vector field representations for navigation, such as SDFs and GDFs. These fields are fed as input to the *Reactive Controller* module, which combines them with other analytic vector fields into a Riemannian Motion Policies formulation (Sec. III-B). Finally, we describe the integration of the reactive controller with our VT&R system in Sec. III-C.
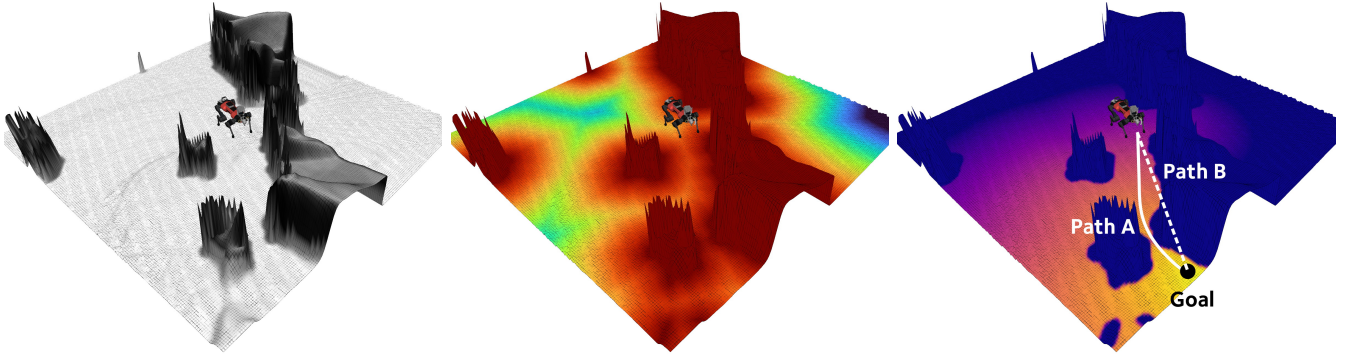
Fig. 3: Representations computed from the elevation map, as described in Sec. III-A. **Left:** Traversability layer computed from geometric analysis; light gray means traversable, whereas black is not traversable. **Center:** Signed Distance Field (SDF); color indicates the distance to obstacles from red (closer) to blue (further). **Right:** Geodesic Distance Field (GDF); the goal is shown in yellow, and further areas in dark blue. Using the GDF, path A has a smaller cost compared to B, which goes over the obstacles.

A. Efficient Vector Field Representations

When navigating over rough terrain with legged robots such as ANYmal C or Spot, 2D representations of the environment are often insufficient. Elevation maps are a popular representation for such applications as they (1) preserve 2.5D information about the terrain, (2) can be easily computed from depth cameras or LiDARs via ray-casting [31], (3) they are now usually available in legged robots since they are used for locomotion tasks, and (4) the elevation map can be effectively treated as an image, making it suitable for efficient computer vision operations.

Thus, we assume that a local elevation map reconstructing a limited space around the robot is available. We base our implementation on the universal grid map library [18], which represents the local elevation map in a multi-layered fixed-sized grid. We use a $8\,\text{m} \times 8\,\text{m}$ map with a resolution of $4\,\text{cm}$, giving a grid of 200×200 cells. To obtain the vector field representations for navigation, we follow the steps explained here:

1) Filling unknown areas: Our first step is to run an in-painting filter on the elevation map. Image-based in-painting methods fill the missing data using nearby pixels. We use the inpainting approach [32] implemented in OpenCV with a radius of $10\,\text{cm}$. Please refer to Sec. IV-B for a further discussion.

2) Traversability computation: We compute a continuous measure for terrain traversability using the slope and roughness of the local map [10]. We additionally take into account the height of terrain with respect to the body pose. This means that if the robot is inclined, e.g. on a hill, the height is measured normal to the surface and the sloped surface is treated as traversable. An example of the continuous traversability map is shown in Fig. 3 left. The continuous value is thresholded to produce a binary classification $t \in [0, 1]$, where 1 means fully traversable and 0 is not traversable.

3) Signed Distance Field (SDF): The SDF layer (Fig. 3 center) is computed using the 2D Euclidean distance transform [33]. Given the binarized traversability layer, we apply the distance transform to obtain the field f_{SDF}. To compute the normalized gradient ∇f_{SDF}, we use the Sobel edge-detector,

which is further smoothed using image filters (Gaussian, median, etc). By relying on 2D image operations, we are able to compute the SDF online at $10\,\text{Hz}$, which is crucial to allow the controller to be responsive to new obstacles.

4) Geodesic Distance Field (GDF): Geodesic distance fields represent a potential function for which a global minima can be computed. This is used to reach a goal from any point in a local map. Similar to the SDF, we use the Fast Marching Method (FMM) [23] on the binarized traversability layer to build the GDF layer. The FMM solves the Eikonal equation [34] from a seed point, from which a wavefront is propagated to fill the entire space. In our implementation, the seed is given by the goal position in the local map or, if the goal falls outside the map, the closest point in the map is chosen. The resulting field is shown in Fig. 3 right. After the field f_{GDF} is computed, we also obtain normalized gradients using the Sobel operator, which defines the *geodesic flow* ∇f_{GDF} to reach the goal.

B. Locally Reactive Controller

In an enviroment without obstacles, the GDF would suffice to generate a motion policy to guide the robot towards the goal. However, just relying on the GDF is insufficient and other priorities such as moving in a certain direction or avoiding obstacles need to be considered.

We combine different tasks specified by tuples $\{\mathbf{f}, \mathbf{M}\}$, also known as a Riemannian Motion Policy (RMP). Here $\mathbf{f} = \mathbf{f}(\mathbf{x}, \dot{\mathbf{x}})$ is an acceleration field that describes a desired behavior and $\mathbf{M} = \mathbf{M}(\mathbf{x}, \dot{\mathbf{x}})$ is a matrix –the *Riemannian metric*– that determines the importance of each acceleration field $\mathbf{f}$. The desired behaviors can be described in terms of goal reaching, obstacle avoidance, damping, etc. Given the set tasks $\{\mathbf{f}, \mathbf{M}\}$, we find the optimal control $\ddot{\mathbf{x}}$ by minimizing:

$$\arg\min_{\ddot{\mathbf{x}}} \sum_{\{\mathbf{f}, \mathbf{M}\}} \|\mathbf{f} - \ddot{\mathbf{x}}\|_{\mathbf{M}}^2. \tag{1}$$

Since the optimization is a linear least-squares problem, it can be solved in closed form, and the desired acceleration is the weighted average of the RMPs.
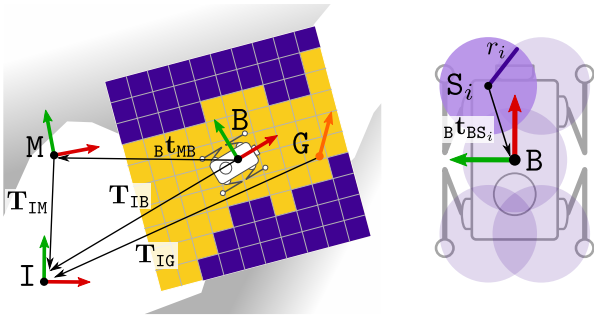
Fig. 4: Reference frames used in this work. **Left:** The pose of the robot's body B in the inertial frame I is given by $\mathbf{T}_{\text{IB}} \in \text{SE}(2)$. The local map, expressed in frame B, represents a local representation of the environment. **Right:** The collision spheres $\mathcal{S}$ define the frame S_i and are used to represent the robot's volume, their positions are expressed in the body frame by the vector $_\text{B}\mathbf{t}_{\text{BS}_i}$.

While the RMP formulation in Eq. (1) is straight-forward, the following aspects must be considered:

1) State representation and reference frames: Our formulation considers the frame conventions illustrated in Fig. 4. The state $\mathbf{x}$ of the robot is given by the pose of the robot's body $\mathbf{T}_{\text{IB}} \in \text{SE}(2)$ expressed in the fixed, inertial frame I. This pose is given by our VT&R system or by other autonomy system. The robot's velocity $\dot{\mathbf{x}}$ and acceleration $\ddot{\mathbf{x}}$ are also expressed in the body frame B and are denoted by $_\text{B}\mathbf{v} = (_\text{B}v_x, _\text{B}v_y, _\text{B}v_\theta) \in \mathbb{R}^3$ and $_\text{B}\mathbf{a} = (_\text{B}a_x, _\text{B}a_y, _\text{B}a_\theta) \in \mathbb{R}^3$ respectively.

The goal pose is expressed in the inertial frame $\mathbf{T}_{\text{IG}} \in \text{SE}(2)$. The local map is computed in the map frame M which has a translation offset of $_\text{B}\mathbf{t}_{\text{MB}} \in \mathbb{R}^2$ to coincide with the center of the robot body.

Note that the reference frame used by the motion policies are described in body frame B, whereas ∇f_{SDF} and ∇f_{GDF} are defined in the local map frame M. These gradients need to be properly transformed to B before being used to solve the optimization problem in Eq. (1). Please refer to the Appendix for technical details.

2) Collision spheres: To represent the robot's volume, we follow the same approach as previous works by using a set of collision spheres $\mathcal{S}$ [9], as shown in Fig. 4, right. Each sphere defines a reference frame S_i, and is parametrized by a fixed translation $_\text{B}\mathbf{t}_{\text{BS}_i}$ and a radius r_i.

Our final formulation computes the optimal acceleration of the body $_\text{B}\mathbf{a}$ given the RMPs summarized in Tab. I.

$$\underset{_\text{B}\mathbf{a}}{\arg\min} \underbrace{\sum_{\{_\text{B}\mathbf{f}, \mathbf{M}\}} \|_\text{B}\mathbf{f} - _\text{B}\mathbf{a}\|^2_\mathbf{M}}_{\text{RMPs applied to body}} + \underbrace{\sum_{\mathcal{S}} \sum_{\{_\text{S}\mathbf{f}, \mathbf{M}\}} \|_\text{S}\mathbf{f} - \mathbf{J}_{\text{SB}}\,_\text{B}\mathbf{a}\|^2_\mathbf{M}}_{\text{RMPs applied to collision spheres}}. \tag{2}$$

In Eq. (2), we expand the original formulation in Eq. (1) to explicitly state where each motion policy is being applied. This can be either the body frame B or the collision spheres $\mathcal{S}$. When applying RMPs to collision spheres, they only affect the center point, and therefore the accelerations only have translational components. Hence, the Jacobian –or *pullback*– $\mathbf{J}_{\text{SB}}$ is required to map the effect of the desired acceleration

of the body $_\text{B}\mathbf{a}$ to the corresponding collision sphere.

The resulting acceleration $_\text{B}\mathbf{a}$ is time-integrated at the controller rate (10 Hz) to obtain a body twist $_\text{B}\mathbf{v}$ which is then passed to the whole-body controller (WBC). We use a WBC, developed by collaborators and based on Lee *et al.* [35], that generates a valid walking motion given a desired velocity using a learned gait model, which is robust over uneven terrains.

C. Integration with the VT&R System

For closed-loop integration with the VT&R system, we use the *carrot-on-a-stick* strategy, which sends a lookahead goal at a distance d_{carrot} which ensures that the robot moves continuously along reference trajectory. Additionally, we implemented status feedback from the controller to the VT&R system, which not only reports when a goal is reached, but also if a goal is unreachable, for instance, when the path is blocked.

IV. RESULTS

We performed experiments both in simulation and on the ANYmal C100 robot with an Intel i7 computer onboard. All the processing was CPU-based. The robot has been modified from the stock version by replacing the front and rear cameras with two RoboSense BPearl LiDARs. The data from the LiDARs was used to build local elevation maps using the approach of Fankhauser *et al.* [31]. Additionally, a front wide-angle stereo camera developed by Sevensense Robotics was installed on top and was used as input for the VT&R system.

We tested our system in a cluttered indoor space, as well as a large-scale underground mine. We evaluated the performance of our system both quantitatively and qualitatively in the two scenarios. In Tab. I we provide a qualitative study of the contribution of individual motion policies, as well as the computation time required by our system.

A. Study 1: Influence of each RMP

In this experiment, we provide a qualitative analysis performed in a simulated environment to understand the influence of the different RMPs considered. In Fig. 5, we show three examples which use different combinations of the RMPs. The robot was given a goal on the other side of the wall with the heading in the same direction as the starting pose. Fig. 5 (a) shows that by using only *Obstacle-free Goal Reaching* + *Obstacle Avoidance* RMPs, the solution falls into local minima as the goal was at the other side of the wall. The robot kept a safe distance from the wall, since most of the collision avoidance metrics are enabled (red circles).

By using the *GDF-based Goal Reaching* and *Obstacle Avoidance* RMPs, the robot is able to go around the wall. However, the final orientation of the robot does not match the goal, having $\approx 180°$ error as seen in Fig. 5 (b), as the GDF-based RMP does not have an orientation component.

In Fig. 5 (c), the weighted combination of RMPs via distance-enabled metrics allows the robot to go around the obstacle by relying on the GDF when the goal is farther

TABLE I: Main Riemannian Motion Policies (RMP) used for the reactive controller formulation.

RMP	GDF-based Goal Reaching	Obstacle-free Goal Reaching	Obstacle Avoidance	Heading Correction	Damping	Regularization
Description	Uses the GDF to reach the goal	Minimizes the position and orientation error assuming free space	Uses the SDF to push the robot away from obstacles	Aligns the robot with the actual translation-only velocity vector	Damps the output acceleration depending on the current velocity	Smooths the output acceleration using the last acceleration command
Applied to	Body B	Body B	Collision sphere $\mathtt{S}_i$	Body B	Body B	Body B
Acceleration field f	$-k\,\nabla f_{\mathrm{GDF}}({}_{\mathtt{M}}\mathbf{t}_{\mathtt{MB}})$	$k\,\mathrm{Log}\left(\mathbf{T}_{\mathtt{IB}}^{-1}\mathbf{T}_{\mathtt{IG}}\right)^{\star}$	$-k\,\dfrac{\nabla f_{\mathrm{SDF}}({}_{\mathtt{M}}\mathbf{t}_{\mathtt{MB}})}{(f_{\mathrm{SDF}}({}_{\mathtt{M}}\mathbf{t}_{\mathtt{MB}})-r_i)}$	$k\,\mathrm{atan2}({}_{\mathtt{B}}v_y,\,{}_{\mathtt{B}}v_x)$	$-k\,{}_{\mathtt{B}}\mathbf{v}$	${}_{\mathtt{B}}\mathbf{a}_{t-1}{}^{\dagger}$
Components affected	Only translation, ${}_{\mathtt{B}}f_\theta = 0$	Translation and orientation	Only translation, ${}_{\mathtt{S}}f_\theta = 0$	Only orientation, ${}_{\mathtt{B}}f_x = {}_{\mathtt{B}}f_y = 0$	Translation and orientation	Translation and orientation
Metric M	$\sigma(d, d_c)\,\mathbf{I}_{3\times3}$ ‡	$\sigma(d, d_c)\,\mathbf{I}_{3\times3}$	$\sigma(d, d_c)\,\mathbf{I}_{2\times2}$	$\sigma(d, d_c)\,\mathbf{I}_{3\times3}$	$\sigma(d, d_c)\,\mathbf{I}_{3\times3}$	$s\,\mathbf{I}_{3\times3}$
When Enabled?	Distance to goal d is **further** than d_c	Distance to goal d is **closer** than d_c	Distance to obstacle d is **closer** than d_c	Distance to goal d is **closer** than d_c	Always	Always

⋆ Log $(\cdot)$ is the logarithm map of SE(2).
† ${}_{\mathtt{B}}\mathbf{a}_{t-1}$ corresponds to the acceleration computed in the previous iteration.
‡ The function $\sigma(d)$ is implemented as a logistic/inverse logistic function that becomes 1 when d is closer/farther than d_c depending on the case.

than a distance d_c, whereas it tries to minimize orientation error using the *Obstacle-free Goal Reaching* RMP within a distance d_c of the goal, where $d_c = 1\,\mathrm{m}$ in our experiments.

Fig. 6: Study 2: Computation time of the filter chain and reactive controller.

B. Study 2: Computation time

We measured the computation time of the filter chain and the controller. We ran our controller on the real robot to collect data for about 5 min during a typical VT&R mission. Fig. 6 shows the computation time across the whole experiment, obtaining an upper bound of 80 ms ($\approx$12 Hz) on average. Most of the computation is spent in the in-painting operation, followed by the GDF computation which requires about 13 ms on average. Using coarser grid resolutions, smaller local maps, or simpler in-painting strategies could help to increase the operation frequency if required.

By comparison, the controller has minimum computational cost, requiring <2 ms in average. Setting up the RMPs and solving the optimization problem also has a negligible cost due to the simplicity of the approach. This could allow us to operate at higher frequency if we reuse the last elevation map processed by the filter chain.

C. Experiment 1: Cluttered Indoor Space

This experiment was designed to demonstrate and evaluate the performance of our VT&R system integrated with the reactive controller in a cluttered indoor environment. We performed a teach step along a short trajectory (18 m) in our lab environment. We placed objects along the route to create an obstacle course to demonstrate (1) the performance of our
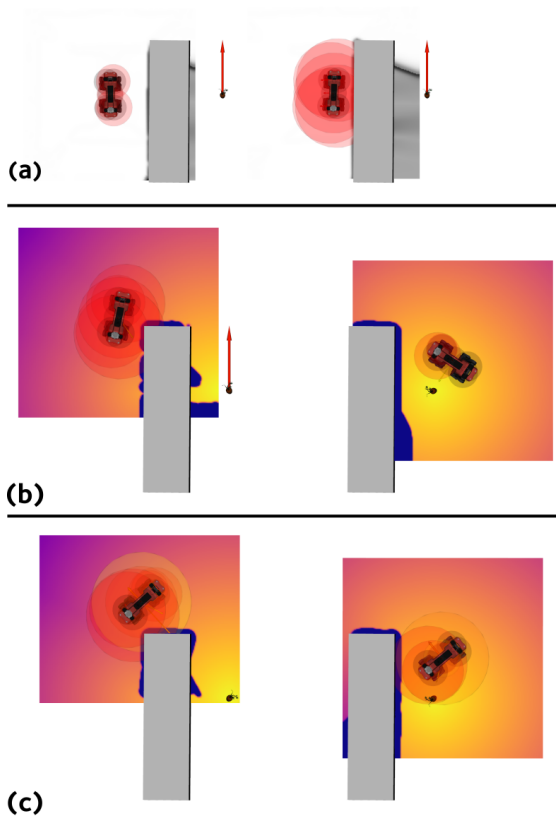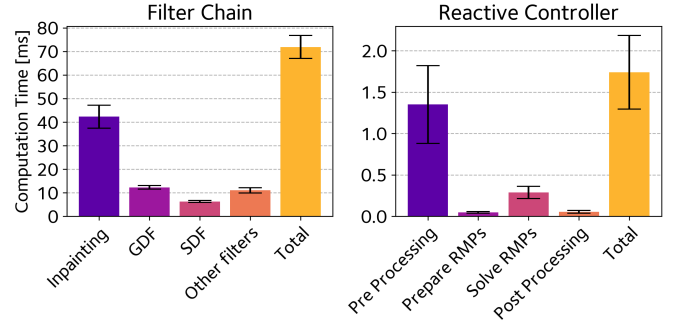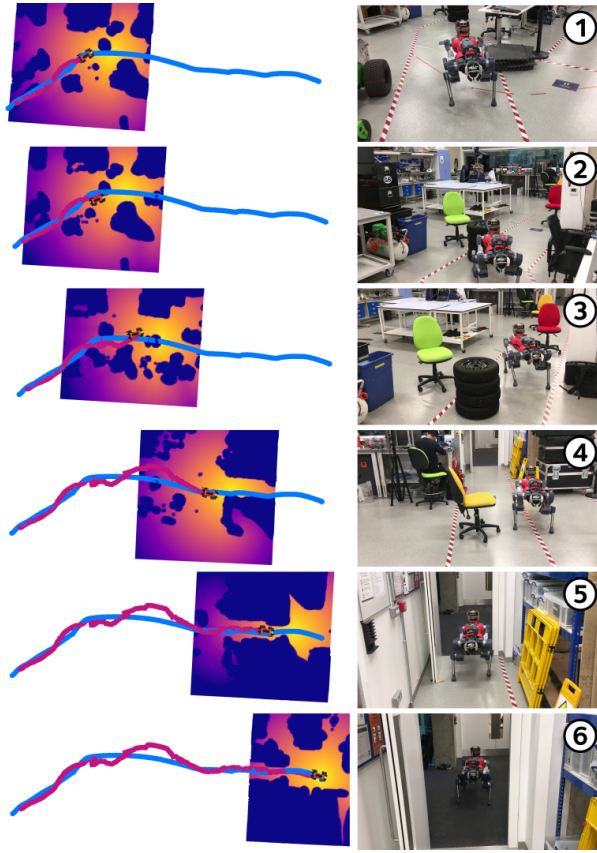

Fig. 5: Study 1: Influence of RMPs. **(a)** *Obstacle-free Goal Reaching + Obstacle Avoidance RMPs*. **(b)** *GDF-based Goal Reaching + Obstacle Avoidance RMPs*. **(c)** *All the RMPs in Tab. I.* The red circles' size is proportional to the importance of the collision avoidance metrics.

Fig. 8: Experiment 1: Path tracking error for the four different traversals in the indoor environment.

Fig. 7: Experiment 1: Indoor Space.**Top:** Example of a repeat traversal (*Forward2*, in magenta ■) after introducing obstacles in the original teach path (in blue ■). **Left:** traversed path as well as the GDF computed online. **Right:** Footage from the actual traversal.

reactive controller in terms of avoiding obstacles, and (2) the robustness of the overall system to deal with degraded appearance of the environment introduced due to obstacles. We ran the system in repeat mode four times (twice walking forwards and twice backwards), moving the obstacles between each run. Fig. 7 illustrates some of the behaviors emerging after introducing obstacles to the reference path.

We computed the *path tracking error* (PTE) to evaluate the tracking performance of VT&R systems [5] shown in Fig. 8. As expected, the tracking performance decreased near the obstacles (indicated with the spikes in the errors, numbered ②, ③, and ④), as the controller forces the robot to deviate from the reference path using the cues given by the GDF as well as the obstacle avoidance policies. In the first run (*Forward1*), we used a carrot distance $d_{carrot} = 0.7$ m resulting in an average speed of 0.16 m/s. Whereas in the other traversals, we used $d_{carrot} = 1.5$ m which allowed the robot to run faster at 0.31 m/s on average.

D. Experiment 2: Large Underground Mine

Our second experiment was executed in an unlit decommissioned underground mine in Wiltshire, UK. In this experiment, we teleoperated the robot along a 60 m long teach path which passed through narrow passages close to walls and large areas that challenged the reliability of stereo vision. The
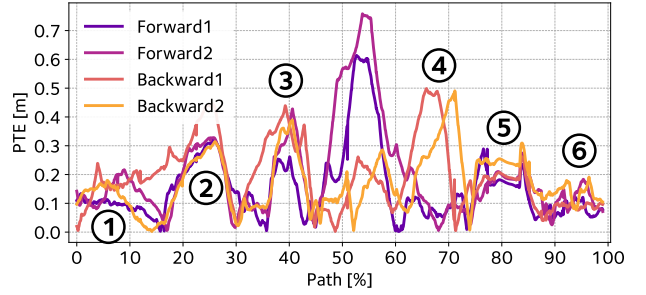
robot carried onboard lights and we enhanced the images using CLAHE equalization [36] to deal with the poor lighting conditions.

In repeat mode, we had the robot traverse the path twice - forwards as well as backwards. Fig. 9 shows the resulting trajectory traversed by the robot (center), as well as some examples of visual tracking from the VT&R system (Fig. 9, a-d on the top). We also illustrate some of the challenging areas that the robot traversed during the repeat phase, and highlight these areas along the trajectory (Fig. 9, ① − ⑧ on the sides). The robot also safely navigated narrow corridors with a width of 60 cm with the opening being only slightly larger (< 30 cm) than the robot body's width, as shown in ⑥.

As for the previous experiment, we also show the PTE in Fig. 10. The robot achieved better tracking when traversing in the same direction (orange) than the reverse traversal (in purple); this is consistent with previous studies of the optimal placement of cameras for visual odometry [37]. The most significant deviations with respect to the reference path were when rounding corners when walking backwards, see locations ⓑ and ⓓ in Fig. 9 and Fig. 10. Here, visual tracking failed causing the system to fall on the propagated estimate using the robot's odometry until recovery. Despite the loss of visual tracking, the robot traversed in both runs without collision even in narrow corridors with an average speed of 0.3 m/s.

E. Limitations and Opportunities

Our experiments demonstrate that by relying on cues from the local map, we can improve the safety of our VT&R system in an efficient way. However, we observed a few limitations:

1) Dealing with dynamic obstacles: Currently, we are unable to deal with highly dynamic obstacles due to the limitations of the elevation mapping system [31]. Since this mapping system is designed for accurate terrain mapping, its probabilistic fusion does not distinguish between moving objects or outliers.

2) Improved traversability: As we compute the traversability map merely using geometric information, our approach cannot be used in certain environments such as forest floors with vegetation where it might be necessary to traverse over regions typically treated as obstacles. We believe that
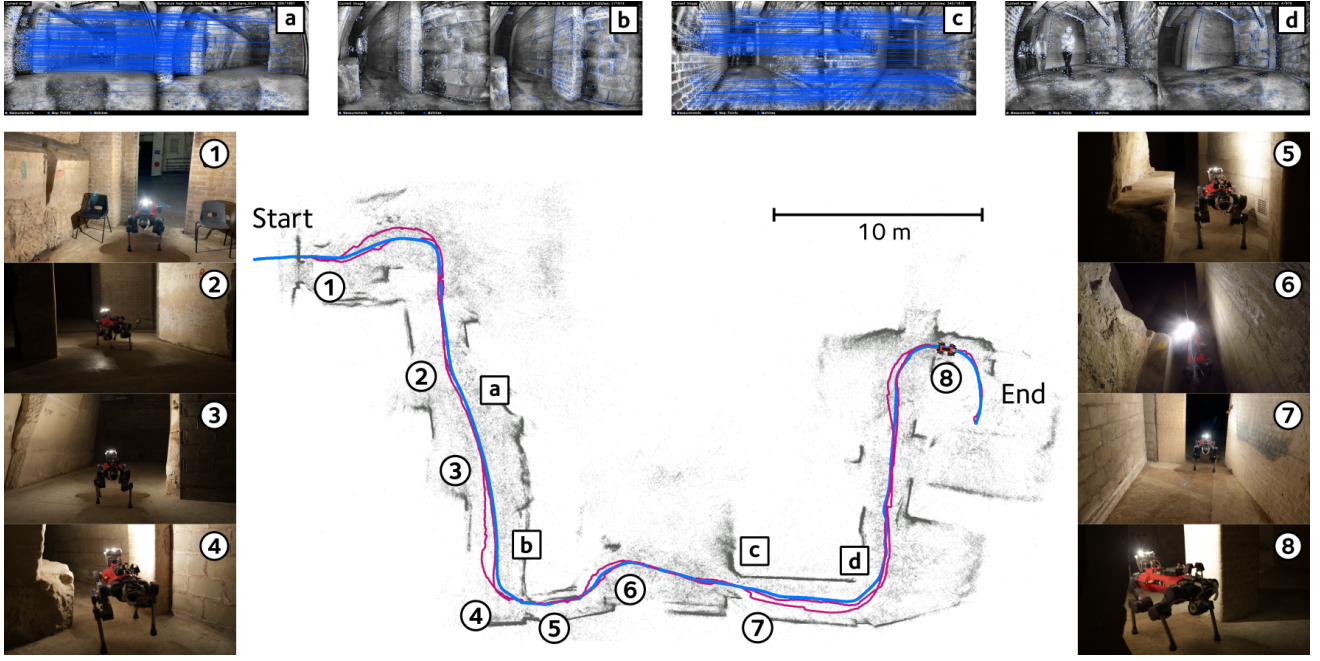
Fig. 9: Experiment 2: Underground Mine. Blue ■ 60 m teach trajectory; Magenta ■ two VT&R-estimated repeat trajectories (forward and backward). The map in gray ■ was not used by the VT&R system and is only shown to illustrate the test area. The pictures on the sides depict the robot executing the mission, while examples of visual tracking are shown on top.
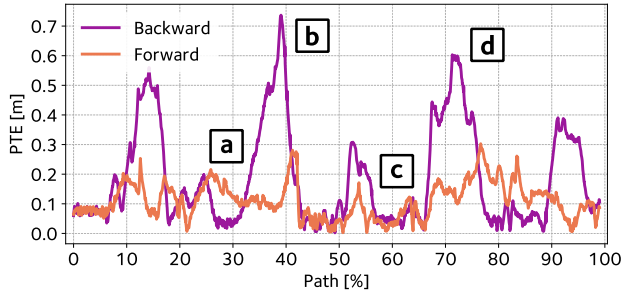

Fig. 10: Experiment 2: Path tracking error obtained in the underground mine for forward and backward traversals.

by incorporating semantic information, similar to some other approaches [21], [19], we could compute a richer traversability metric for other environments.

V. CONCLUSIONS

In this work we presented a locally reactive controller coupled with our VT&R navigation system that allowed safe navigation in challenging environments. Our approach is based on computing efficient vector field representations from a local map, which are used to compute acceleration fields that we combined using a Riemannian Motion Policies controller to generate a twist command at 10 Hz. We deployed our system on an ANYmal robot and showed that our locally reactive controller allowed the robot to traverse a cluttered path taught without obstacles. We also demonstrated the system safely navigating through challenging rooms and corridors of an underground mine, walking at 0.31 m/s in average.

For the next steps, we aim to extend our system to autonomously navigate in 3D environments using perceptive whole-body controllers, as well as dealing with drastic appearance changes using learning-based methods.

APPENDIX
FRAME TRANSFORMATIONS OF ACCELERATION FIELDS

In order to build the *GDF-based Goal Reaching* and *Obstacle Avoidance RMPs*, we need to evaluate the gradients of SDF and GDF to generate acceleration fields in the body frame B. First, we determine the pose of the body in the map frame M as (see Fig. 4):

$$\mathbf{T}_{MB} = \mathbf{T}_{IM}^{-1} \mathbf{T}_{IB} \qquad (3)$$

This can be used to evaluate the ∇f_{SDF} and ∇f_{GDF} at the desired location $_M\mathbf{t}_{MB}$, which corresponds to the translation components of $\mathbf{T}_{MB}$. However, as the gradients are computed in the frame M, we need to transform them to frame B.

The SE(2) adjoint of $\mathbf{T}_{MB}$, denoted by $\mathrm{Ad}\,(\mathbf{T}_{MB})$, maps vectors defined in B to the frame M [38]. Its inverse does the opposite, allowing us to transform vectors defined in M to B:

$$\mathrm{Ad}\,(\mathbf{T}_{MB})^{-1} = \mathrm{Ad}\,(\mathbf{T}_{MB}^{-1}) = \mathrm{Ad}\,(\mathbf{T}_{BM}). \qquad (4)$$

The adjoint is given by:

$$\mathrm{Ad}\,(\mathbf{T}_{BM}) = \begin{bmatrix} \mathbf{R}_{BM} & \mathbf{1}^{\wedge}{}_M\mathbf{t}_{BM} \\ 0_{1\times2} & 1 \end{bmatrix} \qquad (5)$$

where the hat operator $(\cdot)^{\wedge}$ builds a skew-symmetric matrix from the vector, and $\mathbf{R}_{BM}$ and $_M\mathbf{t}_{BM}$ are the rotation and translation components of $\mathbf{T}_{BM}$, respectively.

The gradient ∇f_{GDF} is a vector in $\mathbb{R}^2$ so we can add an extra zero for the angular component to define $\nabla f'_{\text{SDF}} \in \mathbb{R}^3$. Then we use the adjoint to transform the gradients to B:

$$_{\text{B}}\nabla f'_{\text{GDF}} = \text{Ad}\left(\mathbf{T}_{\text{BM}}\right)\nabla f'_{\text{GDF}}. \tag{6}$$

We expand Eq. (6) using Eq. (5) we obtain a simplified expression:

$$_{\text{B}}\nabla f_{\text{GDF}} = \mathbf{R}_{\text{BM}}\nabla f_{\text{GDF}} \tag{7}$$

which basically rotates the gradients to match the frame B.

To evaluate the SDF we follow a similar procedure but adding a transformation defined at the center of the collision sphere at S_i:

$$\mathbf{T}_{\text{MS}_i} = \mathbf{T}_{\text{IM}}^{-1}\,\mathbf{T}_{\text{IB}}\,\mathbf{T}(_{\text{B}}\mathbf{t}_{\text{BS}_i}) \tag{8}$$

with $\mathbf{T}(_{\text{B}}\mathbf{t}_{\text{BS}_i})$ a transformation matrix built from the position of the sphere in the body frame B, $_{\text{B}}\mathbf{t}_{\text{BS}_i}$.

REFERENCES

[1] P. Furgale and T. D. Barfoot, "Visual teach and repeat for long-range rover autonomy," *Journal of Field Robotics*, 2010.

[2] M. Fehr, T. Schneider, and R. Siegwart, "Visual-Inertial Teach and Repeat Powered by Google Tango," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[3] W. Churchill and P. Newman, "Practice makes perfect? Managing and Leveraging Visual Experiences for Lifelong Navigation," *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[4] N. D. Ratliff, J. Issac, D. Kappler, S. Birchfield, and D. Fox, "Riemannian Motion Policies," *CoRR*, 2018.

[5] M. Mattamala, M. Ramezani, M. Camurri, and M. Fallon, "Learning Camera Performance Models for Active Multi-Camera Visual Teach and Repeat," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[6] X. Meng, N. D. Ratliff, Y. Xiang, and D. Fox, "Neural Autonomous Navigation with Riemannian Motion Policy," *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[7] D. Kappler, F. Meier, J. Issac, J. Mainprice, C. G. Cifuentes, M. Wuthrich, V. Berenz, S. Schaal, N. Ratliff, and J. Bohg, "Real-time Perception meets Reactive Motion Generation," *IEEE Robotics and Automation Letters*, 2018.

[8] R. Buchanan, L. Wellhausen, M. Bjelonic, T. Bandyopadhyay, N. Kottege, and M. Hutter, "Perceptive Whole-body Planning for Multilegged Robots in Confined Spaces," *Journal of Field Robotics*, 2021.

[9] M. Gaertner, M. Bjelonic, F. Farshidian, and M. Hutter, "Collision-Free MPC for Legged Robots in Static and Dynamic Scenes," *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[10] M. Wermelinger, P. Fankhauser, R. Diethelm, P. A. Krüsi, R. Siegwart, and M. Hutter, "Navigation Planning for Legged Robots in Challenging Terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[11] D. Hoeller, L. Wellhausen, F. Farshidian, and M. Hutter, "Learning a State Representation and Navigation in Cluttered and Dynamic Environments," *IEEE Robotics and Automation Letters*, 2021.

[12] D. Kim, D. Carballo, J. Di Carlo, B. Katz, G. Bledt, B. Lim, and S. Kim, "Vision Aided Dynamic Exploration of Unstructured Terrain with a Small-Scale Quadruped Robot," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[13] V. Vasilopoulos, G. Pavlakos, S. L. Bowman, J. D. Caporale, K. Daniilidis, G. J. Pappas, and D. E. Koditschek, "Reactive Semantic Planning in Unexplored Semantic Environments Using Deep Perceptual Feedback," *IEEE Robotics and Automation Letters*, 2020.

[14] T. Hines, K. Stepanas, F. Talbot, I. Sa, J. Lewis, E. Hernandez, N. Kottege, and N. Hudson, "Virtual Surfaces and Attitude Aware Planning and Behaviours for Negative Obstacle Navigation," *IEEE Robotics and Automation Letters*, 2021.

[15] P. Krüsi, P. Furgale, M. Bosse, and R. Siegwart, "Driving on Point Clouds: Motion Planning, Trajectory Optimization, and Terrain Assessment in Generic Nonplanar Environments," *Journal of Field Robotics*, 2017.

[16] M. Brandão, O. B. Aladag, and I. Havoutis, "GaitMesh: Controller-Aware Navigation Meshes for Long-Range Legged Locomotion Planning in Multi-Layered Environments," *IEEE Robotics and Automation Letters*, 2020.

[17] S. Pütz, T. Wiemann, M. Kleine Piening, and J. Hertzberg, "Continuous shortest paths vector field navigation on 3D triangular meshes for mobile robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[18] 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*, A. Koubaa, Ed., 2016.

[19] D. M. Bradley, J. K. Chang, D. Silver, M. Powers, H. Herman, P. Rander, and A. Stentz, "Scene understanding for a high-mobility walking robot," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[20] L. Nardi and C. Stachniss, "Actively Improving Robot Navigation on Different Terrains Using Gaussian Process Mixture Models," *IEEE International Conference on Robotics and Automation*, 2019.

[21] L. Wellhausen, A. Dosovitskiy, R. Ranftl, K. Walas, C. Cadena, and M. Hutter, "Where Should I Walk? Predicting Terrain Properties From Images Via Self-Supervised Learning," *IEEE Robotics and Automation Letters*, 2019.

[22] H. Oleynikova, A. Millane, Z. Taylor, E. Galceran, J. Nieto, and R. Siegwart, "Signed Distance Fields: A Natural Representation for Both Mapping and Planning," *Robotics: Science and Systems*, 2016.

[23] J. A. Sethian, "A Fast Marching Level Set Method for Monotonically Advancing Fronts," *Proceedings of the National Academy of Sciences*, 1996.

[24] K. Crane, C. Weischedel, and M. Wardetzky, "The Heat Method for Distance Computation," *Commununications of the ACM*, 2017.

[25] A. Valero-Gomez, J. V. Gomez, S. Garrido, and L. Moreno, "The Path to Efficiency: Fast Marching Method for Safer, more Efficient Mobile Robot Trajectories," *IEEE Robotics and Automation Magazine*, 2013.

[26] S. Ahmad, A. B. Mills, E. R. Rush, E. W. Frew, and J. S. Humbert, "3D Reactive Control and Frontier-Based Exploration for Unstructured Environments," *arXiv*, 2021.

[27] J. Mainprice, N. Ratliff, M. Toussaint, and S. Schaal, "An Interior Point Method Solving Motion Planning Problems with Narrow Passages," in *IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 2020.

[28] C. J. Ostafew, A. P. Schoellig, and T. D. Barfoot, "Robust Constrained Learning-based NMPC enabling reliable mobile robot path tracking," *International Journal of Robotics Research*, 2016.

[29] L. P. Berczi and T. D. Barfoot, "Looking High and Low: Learning Place-dependent Gaussian Mixture Height Models for Terrain Assessment," *IEEE International Conference on Intelligent Robots and Systems*, vol. 2017-September, 2017.

[30] S. R. Bista, B. Ward, and P. Corke, "Image-Based Indoor Topological Navigation with Collision Avoidance for Resource-Constrained Mobile Robots," *Journal of Intelligent and Robotic Systems: Theory and Applications*, 2021.

[31] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic Terrain Mapping for Mobile Robots With Uncertain Localization," *IEEE Robotics and Automation Letters*, 2018.

[32] A. Telea, "An Image Inpainting Technique Based on the Fast Marching Method," *Journal of Graphics Tools*, 2004.

[33] P. F. Felzenszwalb and D. P. Huttenlocher, "Distance Transforms of Sampled Functions," *Theory of Computing*, 2012.

[34] G. Peyré, M. Péchaud, R. Keriven, and L. D. Cohen, "Geodesic Methods in Computer Vision and Graphics," *Foundations and Trends in Computer Graphics and Vision*, 2010.

[35] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, 2020.

[36] K. Zuiderveld, "Contrast Limited Adaptive Histograph Equalization." in *Graphic Gems IV*. Academic Press Professional, 1994.

[37] V. Peretroukhin, J. Kelly, and T. D. Barfoot, "Optimizing Camera Perspective for Stereo Visual Odometry," in *Canadian Conference on Computer and Robot Vision*, 2014.

[38] J. Solà, J. Deray, and D. Atchuthan, "A micro Lie theory for state estimation in robotics," *arXiv*, 2018. [Online]. Available: http://arxiv.org/abs/1812.01537