Ground Vehicle Navigation through Traversability Analysis of Outdoor Environments

UNIVERSITÀ
degli STUDI
di CATANIA

Dario Calogero Guastella

Advisor: Prof. Giovanni Muscato

Department of Electrical, Electronics and Computer Engineering
Università degli Studi di Catania

Doctor of Philosophy
November 2018

Acknowledgements

I wish to thank my supervisor Professor Giovanni Muscato for his valuable support and guidance throughout the last three years. He gave me the opportunity to have enriching and inspiring experiences, both from the research and from the human perspective.

I would like to thank all my colleagues of the robotics lab, Dr. Luciano Cantelli, Dr. Donato Melita, and Dr. Domenico Longo. Without them, most of the research activity reported in this thesis would not have been possible. I would also like to thank Dr. Filippo Bonaccorso, who, unconsciously, lead me to decide to start my PhD course, and Dr. Valentina Gambuzza, for her valuable advice and the profitable discussions. I wish to thank all the people I met in the lab and my PhD colleagues for both the intriguing and the funny times spent together.

Thank you to my friends, Andrea, Nuccio, Giuseppe Callea, Giuseppe Caruso, and Vito, for supporting me during these years of intensive effort. Thank you to my family, my brothers and parents for their understanding, especially in the last months. A special thank to my brother Ivano for his help in proofreading this thesis.

Abstract

This thesis describes the research activity carried out on the navigation of unmanned ground vehicles in outdoor unstructured environments. These environments are very common in a wide variety of real-application scenarios, such as search and rescue for post-disaster response, monitoring of the environment and industrial facilities, precision farming, planetary exploration etc.. For this reason, robotics researchers have been investigating such kind of environments for decades. However, the problem of vehicle navigation in these scenarios is not fully solved. In fact, a *general* approach for ground vehicle motion planning, taking into account both the robot and the environment features, is still far to be defined. In the literature, this problem has been addressed through the so-called *traversability analysis*. It can be seen as an assessment of the difficulty for a specific ground vehicle, characterized by its own locomotion features, to cross a terrain area, which is in turn characterized by its own morphology and appearance.

The works reported in this thesis are related to the problem of geometry-based traversability analysis. It consists in deriving maps of traversing costs from three-dimensional models of the environ-

ment. These costmaps are extremely useful in challenging environments, as they are used in robot motion planning to avoid unsafe paths for the vehicle itself. Still in the context of 3D reconstructions, a solution for the remote drive of mobile robotic platforms has been developed. The aim has been to enhance the operator interface with helpful information, including traversing costs, via graphical elements presented in a mixed reality context. Finally, coverage path planning for unmanned aerial vehicles has been investigated as well. It is a specific kind of path planning related to three-dimensional photogrammetric reconstruction. In particular, an approach to manage a flock of aerial vehicles has been developed, in order to parallelize the coverage mission. The coverage task is the first step needed for the environment reconstruction. Eventually the traversability analysis can be performed on the obtained reconstruction.

Experimental and on-field trials have also been performed on a real mobile robotic platform, for the testing of the proposed approaches. The results achieved are reported and widely discussed.

Contents

List of acronyms

Acronyms / Abbreviations

AV	Augmented Virtuality
CCPP	Complete Coverage Path Planning
DEM	Digital Elevation Model
DNN	Deep Neural Network
DoF	Degree of Freedom
ENU	East-North-Up
GIS	Geographic Information Systems
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HMD	Head-Mounted Display
IMU	Inertial Measurement Unit

IRL Inverse Reinforcement Learning

MR Mixed Reality

PCA Principal Component Analysis

RGB-D Red Green Blue and Depth

ROI Region Of Interest

ROS Robot Operating System

RTK Real-Time Kinematic

UAV Unmanned Aerial Vehicle

UGV Unmanned Ground Vehicle

UTM Universal Transverse Mercator

VA Visual Aid

VR Virtual Reality

List of Figures

List of Tables

Chapter 1

Introduction

When human beings move within an environment, they are quickly capable to decide which way prefer or not. In doing that, they take into account several aspects, such as the presence of obstacles or, in general, the hindrance that may be faced by choosing a certain path rather than another one. This process is so immediate and natural for our cognitive system that one could erroneously think it is a *simple* process as well. However, these kinds of assessments, trivial for a human being, do represent relevant problems for a robotic system. To date, defining a feasible motion for a ground vehicle, as a result of the observation and the "analysis" of the surrounding environment, is still an open issue. In the literature, this kind of problem is typically referred to as *traversability analysis*. Most of times it is more specifically referred to as *terrain* traversability analysis or, occasionally, as drivability, trafficability, navigability, coverability, terrainability [72]. Throughout the present thesis the term "terrain" will be used to indicate, in general, the ground the vehicle has to cross, regardless of

whether roads, rocks or soil are being considered. This is because the reference environments considered in traversability analysis are mainly *outdoor natural environments*, although most of the times the type of assessment performed can be indistinctly applied to urban or even indoor environments.

1.1 Motivation

Nowadays the importance of autonomous navigation, and, as a consequence, of traversability analysis, is further increased due to the wide diffusion over the last decades of mobile robots, both in service and industrial robotics. Furthermore, ground mobile robots can be applied to a wide spectrum of sectors, ranging from planetary explorations to search and rescue operations, precision agriculture, urban, domestic, working, environmental and industrial application, to name a few. Reasonably, all these areas could be considerably impacted by the research activity carried out on this branch of robotics.

In general, whenever human intervention is extremely hard or dangerous, if not even unrealizable or harmful, the adoption of *Unmanned Ground Vehicles* (*UGV*s) does represent an important opportunity, which opens to scenarios and possibilities never explored before. These situations, often characterized by the most difficult and arduous environments, offer the most challenging goals for robotics research about ground vehicles. This kind of environments are typically referred to as *unstructured*, meaning that they are lacking in useful landmarks or clearly viable paths. This makes extremely com-

plex to plan safe trajectories for a UGV, since both robot stability and path execution feasibility must be guaranteed.

1.2 Problem statement

According to [72], the term *traversability* refers to the capability of a ground vehicle to reside over a terrain area, under an admissible state, moving from its current state. Such *capability* is a function of the following elements: the terrain, the robotic vehicle and its kinematic constraints and, optionally, a set of optimization criteria. Understandably, both the terrain and the vehicle *models* adopted become crucial in this kind of problem. Since traversability is meaningful as the vehicle has to *move towards* a new terrain area, it is also commonly referred to a *traversing cost*, which recalls the idea of difficulty in crossing the considered area.

In [54], the traversing cost for a mobile robot is more formally defined as a function of the terramechanical properties, the geometry of the terrain and the robot state

$$\tau(\mathbf{x}) = f(M(\mathbf{x}, t), G(\mathbf{x}, t), X(t)) \tag{1.1}$$

where $\tau(\mathbf{x}) \in \mathbb{R}_+$ is the traversing cost at coordinates $\mathbf{x} \in \mathbb{R}^3$ in the 3D environment; $M(\mathbf{x}, t) \in \mathscr{M}$ is the set of terramechanical properties of the terrain at coordinate $\mathbf{x}$ and time t; $G(\mathbf{x}, t) \in \mathscr{G}$ is the set of geometric features of the terrain at coordinate $\mathbf{x}$ and time t; $X(t) \in \mathbb{R}^n$ is the state of robot model, with n degrees of freedom at time t. The function $f(\cdot)$ is a model mapping all the considered elements to the

traversing cost and its identification can be viewed as the final goal of the terrain traversability analysis problem.

In order to face this problem several kinds of approaches have been presented in this research field, by taking into account different aspects of the area under inspection. Such aspects are essentially related to the terms of function $f(\cdot)$ in the Eq. 1.1. However, most of the solutions available in the literature can be divided into two broad categories [72]:

- *geometry-based approaches*
- *appearance-based approaches*

In geometrical approaches, which are the most widely explored in robotics research, the geometric features of the environment are considered. With respect to Eq. 1.1, this is equivalent to assume the terrain to be homogeneous in its terramechanical properties. Moreover, for the sake of simplicity, both terramechanical and geometrical properties of the terrain are typically assumed to be constant over time. Finally, by considering traversability of terrain independent from the robot state, it is approximate to a function of terrain shape only (i.e. terrain geometry only). Some examples of geometry-based approaches are reported in [10, 65, 95].

$$\tau(\mathbf{x}) \approxeq f(G(\mathbf{x})) \tag{1.2}$$

The second kind of approach in traversability analysis consists, instead, in identifying and classifying the type of terrain to cross [44, 90, 100]. Typically, a classification is performed on a finite set of terrain types, often natural ones, which are defined according to

the their appearance (namely pattern or color). Therefore, a classification is essentially used to relate appearance to terramechanical properties, and, then, to terrain traversability.

Besides the two main research streams presented, there are several *hybrid* solutions, exploiting the strengths of both methods, that, in fact, show a certain degree of complementarity [51, 85]. Both terrain shape and appearance can be acquired before actually crossing the considered region. In this case Eq. 1.1 turns into

$$\tau(\mathbf{x}) \approxeq f(G(\mathbf{x}), A(\mathbf{x})) \tag{1.3}$$

where $A(\mathbf{x}) \in \mathbb{A}$ is the *appearance* of the terrain at coordinate $\mathbf{x}$.

Recently, a novel research stream exploiting *Deep Neural Networks* (*DNN*s) is giving a renewed momentum to the adoption of *robot learning* solutions [38, 40], which have been already explored in the past [35, 84]. A relevant example is reported in [101] where the *Inverse Reinforcement Learning* (*IRL*) [1] paradigm is combined to a *Convolutional Neural Network* (*CNN*) for non-linear function approximation. IRL belongs to the wider class of *Learning from Demonstration* methods and aims at inferring the so-called *reward function*. Such function provides a kind of description of the *motivations* behind the expert choices and actions. In [101] the CNN is trained in order to approximate the reward function, also called cost function, which essentially represents the traversability costmap learned from the demonstrations by the expert. From another perspective, the CNN is trained to learn the $f(\cdot)$ function mapping terrain geometry to traversing costs (Eq. 1.2). Other works adopted CNNs to learn the direct mapping from terrain appearance (i.e. from images acquired

by an onboard camera) to control actions [8, 98]. However, this approach cannot be formally considered a traversability analysis, as no traversing cost is assessed.

All the definitions given and the approaches described above suggest how *traversability analysis* is conceptually far from the classical *obstacle avoidance* paradigm [54, 72]. In fact, since the early solutions of mobile robot navigation, the environment has been represented as either *free* or *occupied by obstacles* [24]. This kind of *binary* representation can be found in a wide variety of works in the robotics literature. Traversability analysis can be viewed as an *extension* of such representation. Traversability assessment searches for a *continuous measure* (i.e. the traversing cost), as opposed to the discrete binary classification. In terms of machine learning, this may be seen as a *regression problem*, i.e. the problem of predicting a continuous quantity, as opposed to the classification problem, which consists in predicting a label among a finite set of class labels.

1.3 Research objectives and contributions

The main goal of the research activity reported in this thesis is the navigation of unmanned ground vehicles in outdoor unstructured environments. To this end, both fully autonomous solutions and teleoperation approaches have been considered. Furthermore, other related research areas, such as the support to rover navigation by *Unmanned Aerial Vehicles* (*UAV*s) and the management of UAV flocks have been explored.

A contribution to the problem of geometry-based traversability analysis is reported in "*A Global Path Planning Strategy for a UGV from Aerial Elevation Maps for Disaster Response*" presented at the 9th International Conference on Agents and Artificial Intelligence [33]. It describes a workflow for deriving traversability costmaps from *Digital Elevation Models* (*DEM*). The main novelty in this work is the reconstruction of such models from a prior aerial photographic survey, autonomously performed by a multi-rotor. Moreover, the traversability assessment is performed by taking into account also the rover navigation features. In the work presented at the 20th International Conference on Climbing and Walking Robots [14], further results of the terrain traversability analysis previously proposed have been reported, by testing it on larger environments with different features.

A contribution is also given to the problem of remotely driving a ground vehicle. In fact, some outdoor scenarios may prevent the adoption of a fully autonomous strategy, for instance when there is the need to safeguard the vehicle by absolutely avoiding accidents. In all of these cases the role played by the teleoperation interface is crucial. In the work developed, a novel interface exploiting augmented virtuality for sensory-informed navigation is proposed.

The main novelty lies in the simultaneous combination of several features, which have been proposed in the related literature in different teleoperation interfaces. The most relevant features are:

- stereoscopic camera acquisition from the real remote environment

- virtual context combining 3D reconstruction of the driving area, acquired video frames and visual aids for vehicle navigation
- immersive visualization through modern virtual reality headset

Among the suggested visual aids, the information about the traversability assessment described in [33] and the consequent recommended path have been also included, in order to guide the driver through safe and easy to traverse terrain areas. This information is extremely useful in the off-road context. The design rules adopted in the overall interface development leverage the recommendations and the experiences of several works in the teleoperation research.

Finally, the path planning of unmanned aerial vehicles has been investigated as well. First, a 3D path planning and task assignment solution for a team of UAVs has been developed as described in the paper entitled "*3D path planning for UAV swarm missions*", and presented at the 3rd International Conference on Mechatronics Systems and Control Engineering [34]. This work has been progressively integrated within a *Complete Coverage Path Planning* (*CCPP*) framework. This kind of path planning problem is crucial for *photogrammetric reconstruction*, which, in turns, provides the input of the geometrical traversability analysis approach presented in [33]. The overall strategy has been described in a work entitled "*Coverage path planning by swarm of UAVs for UGV traversability analysis*" presented at the 21th International Symposium on Measurement and Control in Robotics [15].

Only few works in the literature propose the adoption of a multi-UAV solution for CCPP: this is the main contribution of the paper mentioned. Furthermore, the proposed framework integrates the main features recommended in this kind of applications by the related literature, such as:

- no-fly zones management via decomposition
- optimal path choice within each subregion according to power consumption and geometry consideration
- negotiation for subregion/UAV assignments
- geographic information system management

The developed framework is one of the few available in the literature integrating all the elements listed above in a stand-alone solution.

1.4 Chapter outline

The remainder of the present thesis is organized as follows.

Chapter 2 recalls the fundamentals of the general problem of navigation, from robot kinematics to environment representations. The rover adopted for most of the testing performed throughout this thesis is also presented, by describing the detail of its control system architecture.

Chapter 3 reports an approach for the autonomous navigation of a ground vehicle based on the terrain traversability analysis described in [14, 33]. The assessment performed takes into consideration the environment geometry, obtained by photogrammetric

reconstruction performed by a UAV, and the rover navigation features. The implementation and the tests on a real platform are presented.

Chapter 4 describes the novel control station interface for ground vehicle teleoperation, mentioned in the previous section. First, a wide literature review on interfaces for telerobotics in different application environments is provided. After that, the designed system is described, by focusing on the most relevant features and comparing them with the other works available in the related literature.

Chapter 5 covers the problem of complete coverage path planning. A particular attention is paid on the solutions for aerial vehicles, both as individual agents and as members of a team. First, an approach for CCPP of flocks of aerial vehicles is theoretically described. After that, several experimental tests are discussed, carried out with the help of the software framework implementing the conceived strategy. This chapter is related to the works reported in [15, 34]

Finally, **Chapter 6** summarizes all the results described in previous chapters, providing an overview of the three-year long research activity carried out. Furthermore, an outlook on future ideas to develop and some enhancements to the already developed solutions are proposed.

Chapter 2

Mobile robot navigation

In this chapter the fundamentals of the general problem of mobile robot navigation are recalled. It is the most relevant issue in *mobile* robotics and it involves several aspects. These aspects are the building blocks of a *see-localize-plan-act cycle* (Figure 2.1) [83]. As follows, it can be seen as a very high-level *feedback control scheme* for mobile robotic systems. Four main building blocks can be identified:

1. localization
2. path planning
3. motion control
4. perception

Each of these blocks answers to one of the four key *questions* in autonomous mobile robotics: *where am I?, how do I go from a point to another, while avoiding obstacles?, how do I actually execute the path?, how do I interpret raw data coming from sensors?*.

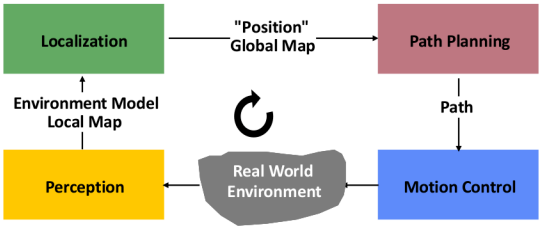


Figure 2.1 The see-localize-plan-act cycle in robotics navigation (image from [83] ©2011, MIT Press)

In the following sections, a general overview of some of these aspects is given, with the aim of providing the reader with the basics to comprehend the subsequent chapters. Finally, the chapter concludes with a section presenting the ground vehicle adopted for testing in most of the works reported in this thesis.

2.1 Mobile robot kinematics

The general problem of mobile robot kinematics is very wide and tightly related to the specificity of the robotic platform considered. A robotic system is first of all a mechanical system, with its own locomotion features and kinematic constraints. In this section the kinematics of a basic *wheeled* robot configuration, the so-called *differential drive* robot, is discussed. It has two driving wheels with collinear axes of rotation tied to a rigid robot chassis, which is usually supported by a passive omnidirectional or caster wheel (Figure 2.2). In the first instance, the robot is supposed to move on an horizontal plane: then its *pose* is completely defined by its *2D position* on

the plane and the robot *heading*. This means that the vehicle has 3 *Degrees of Freedom* (*DoF*), clearly neglecting possible internal degrees of freedom related to further wheel joints (in caster wheels for instance).

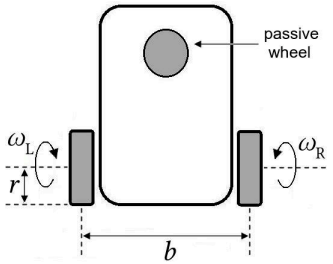


Figure 2.2 The differential drive robot

The first step to formally study platform kinematics is to define two reference frames: the *global reference frame* of the plane, O-$x_I y_I$, which is a static inertial frame, and the *vehicle reference frame*, rigidly attached to the robot and whose origin is located at an arbitrary point of the vehicle chassis, O_R-$x_R y_R$ (Figure 2.3). In the differential drive case the origin is usually placed at the midpoint between the wheels. Moreover, both global and vehicle reference frames are right-handed, with the latter usually chosen in such a way the x_R-axis is along the forward direction of the vehicle, the y_R-axis is on the left and z_R-axis points up.

Robot position is defined as the (x, y) coordinates of the origin O_R in the global reference frame, while vehicle orientation is given by the relative rotation θ between global and local reference along the z_I-axis (which is orthogonal to the plane). Therefore, robot pose

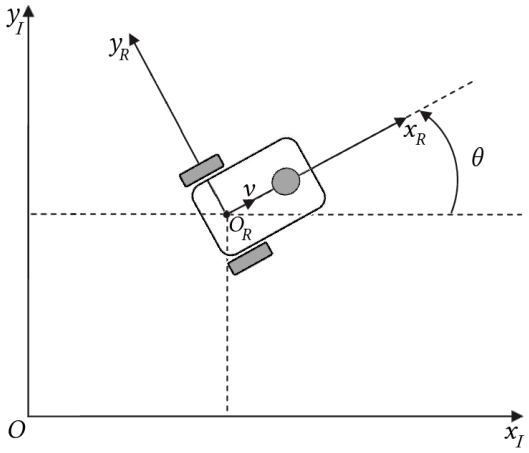


Figure 2.3 Global reference frame O-$x_I y_I$ and vehicle reference frame O_R-$x_R y_R$

is described by the following three-element vector (in the global reference frame)

$$\xi_{\mathbf{I}} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \tag{2.1}$$

By indicating the vehicle linear and angular velocities respectively with v and ω, the following relationships can be derived

$$\begin{aligned} \dot{x} &= v\cos\theta \\ \dot{y} &= v\sin\theta \\ \dot{\theta} &= \omega \end{aligned} \tag{2.2}$$

In the case of the differential drive v and ω can be easily related to the angular velocities of each wheel, ω_L and ω_R. If b denotes the *wheelbase*, i.e. the distance between the two wheels and r the radius of both wheels (see Figure 2.2), the following relationships hold:

$$\begin{aligned} v &= \frac{r(\omega_R + \omega_L)}{2} \\ \omega &= \frac{r(\omega_R - \omega_L)}{b} \end{aligned} \tag{2.3}$$

By substituting Eq. 2.3 into Eq. 2.2 the *forward kinematics model* of the mobile robot is derived, which relates robot's speed in the global reference frame to robot geometry and the speeds of its wheels. This model allows to estimate future poses of the robot, once wheels' angular velocities are known.

Considering now the case of a robot navigating in a *three-dimensional space*, the third dimension of the global reference frame O-$x_I y_I z_I$ is also taken into account. However, robot pose in this case is slightly more complex to be represented. In fact, while the vehicle position will be simply the 3D vector $\mathbf{p_I} = [x \quad y \quad z]^T$ of the coordinates of O_R (origin of vehicle reference frame), this cannot be simply extended to the vehicle attitude representation. In particular, while vector $\mathbf{p_I}$ provides the *translation* between the global and the robot reference frame, vehicle attitude is essentially the *relative rotation* between such frames. There exist several manners to represent a relative rotation between two reference frames in algebra. Some of them are mainly adopted in robotics and aeronautics. In this section only *Euler angles* (and in particular *roll, pitch and yaw* angles) and *quaternions* will be briefly recalled.

Euler angles are a set of three angles describing the orientation of a vehicle as a sequence of three *coordinate rotations* (i.e. rotations about a single coordinate axis). Although many different coordinate rotation sequences can be considered (twelve in total), the most commonly used is the *ZYX*-convention. Under this convention the overall vehicle attitude is described by the sequence of coordinate rotations of the robot frame in the following order:

1. a rotation of ψ about z-axis
2. a rotation of θ about y-axis
3. a rotation of ϕ about x-axis

The three angles ϕ, θ, and ψ are respectively called *roll*, *pitch* and *yaw* angles. In this case the positive direction of the robot-fixed x-axis is assumed to be along the forward direction of the vehicle, with the y-axis to the right, and the z-axis downward (Figure 2.4).

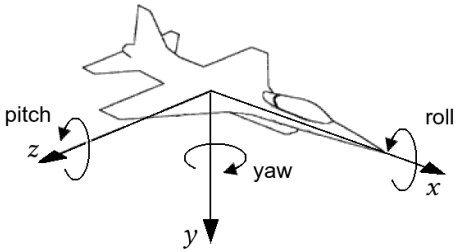


Figure 2.4 The vehicle frame considered in roll, pitch and yaw angles definition

Euler angles are very intuitive to represent orientations and very easy to use. However, they suffer for *singularities* which arise from

the so-called *gimbal lock*. In order to overcome these problems, a further representation was introduced in the literature, based on *quaternions*. A quaternion $\mathbf{q}$ is a four dimensional vector: one component is a scalar number, and the other three define a vector in the imaginary *ijk* space.

$$\mathbf{q} = w + \mathbf{i}x + \mathbf{j}y + \mathbf{k}z \tag{2.4}$$

Attitude of a rigid body is usually represented by *unit quaternions*, i.e. quaternions with unity norm. Unit quaternions are the *minimal*, non-singular description of orientation in 3D space. Besides being non-singular, this representation allows to employ a straightforward algebra to combine relative rotations. Although they are not as intuitive as roll, pitch and yaw angles, unit quaternions have been adopted as attitude representation in our work, since they can be interchangeably applied to either ground or aerial vehicles.

2.2 Localization

Another crucial part of a robotic navigation system is localization. Localization allows the vehicle to know its current position within the environment representation considered. It is fundamental for the robot to verify if it is correctly moving towards the desired goal or not. In the case of *continuous representation* of the environment, localization can be considered as a special case of the transformation between the two reference frames, namely the global frame and the vehicle frame. In fact, "localizing" a vehicle means essentially to define the relative transformation between a *local reference frame*, i.e.

the frame of a *local map* of the environment (i.e. a representation of the environment), and the robot reference frame [92]. Since outdoor environments are the main reference scenario of the present thesis, *GPS-based localization* is discussed in this section.

The *GPS* (*Global Positioning System*) is a *Global Navigation Satellite System* (*GNSS*) allowing to accurately determine position and time (*Universal Time Coordinated, UTC*) anywhere on Earth. In order to determine the position, a GNSS follows a basic principle. Satellites with known positions transmit regular time signals, while the vehicle to be localized carries a satellite signal receiver. This device has a clock that is synchronized to the transmitter's clock. In this manner, the travel time of the radio waves can be measured and, from this value, the position of the receiver from the satellite can be derived, since the electromagnetic signals speed is known. Clearly, such value is the distance from one satellite only; therefore, the vehicle could lie on any point of the surface of a sphere. To unambiguously determine one point at least the distances measured from three satellites must be considered. In practice, in order to mitigate the error related to possible unsynchronized onboard clocks, GNSSs ensure a line-of-sight contact to at least 4 satellites from any point on the ground, thanks to their satellite constellation.

A technique allowing to greatly increase the accuracy of *GPS* localization is the *differential correction*. It involves the adoption of a receiver at a known location, which acts as a *base station*. Once the position of the reference station is exactly known, the deviation of the measured position from the actual position, for each satellite, can be calculated and these corrections are transmitted to the GPS receiver

onboard the vehicle. By combining such differential technique to the *phase measurements* of the carrier signal wave, it is possible to achieve positioning accuracies within centimeters. This solution is called *Real-Time Kinematic* (*RTK*) positioning technique or, specifically in the GPS case, *carrier-phase enhancement* (CPGPS). RTK has been widely adopted in the context of outdoor robotics navigation for years [64, 68].

However, satellite navigation systems have some application limits. First, they require a line-of-sight contact with satellites, which prevents their working in indoors. In general, whenever there are obstructions such as trees, buildings, or natural formations the overall satellite localization quality is worsened. The second main source of error in satellite measurements is the *multi-path problem*. It is caused by local reflections of the GPS, which corrupt the desired signal.

2.2.1 Geographic coordinate systems

The adoption of GPS localization implies to handle *geographical coordinates*, which are referred to specific *geographic coordinate systems*. There are several ways to express the geographical position of a point over the Earth surface. The most commonly adopted are *longitude* and *latitude coordinates*, which however, are not so intuitive and easy to manage since they are angles. A widely used alternative in robotics is the *Universal Transverse Mercator* (*UTM*) projection, since it allows to work with 2D maps. In the UTM system, the Earth surface is divided into sixty *zones*, which are six-degree wide longitudinal bands, and a transverse Mercator projection for each zone is performed.

Therefore, a general location on the Earth surface is given by the UTM zone number and the easting and northing planar coordinates couple in the considered zone, expressed in meters. The origin of the 2D coordinate frame of each UTM zone is located at the intersection of the equator and the zone's central meridian.

Throughout this thesis, local maps are associated to *ENU* frames (*East-North-Up*), by using UTM easting and northing coordinates (x and y coordinates respectively). With regard to the z coordinate, *ellipsoidal heights* have been considered, which are provided by the GPS receiver. These heights are referred to the *World Geodetic System* ellipsoid (*WGS84*), a standard spheroidal reference surface used in geodesy and satellite navigation.

2.3 Path planning

The problem of motion planning can be stated as follows: given a *start pose* of the robot, a desired *goal pose*, a geometric description of the *robot*, and a geometric description of *the world*, define a path that drives the vehicle from start to goal while avoiding the contact with any obstacle. Sometimes this problem is divided into *global* and *local* planning task: the global planner computes approximate paths neglecting the kinematic and dynamic vehicle constraints, whereas the local planner takes into account such constraints and generates feasible local trajectories, while avoiding obstacles.

More formally, a robot pose can be seen as a *configuration* of the system. The *configuration space* $\mathscr{C}$ is the space of all possible robot configurations and the dimensionality of the configuration space is

given by the number of degrees of freedom of the vehicle. A general path planning problem is to find a continuous mapping $c : [0,1] \rightarrow \mathscr{C}$, such that no configuration along the path implies a robot-obstacle collision [20]. The unit interval where the c mapping is defined is the *arc length parameterization* of the planned path.

In the literature, several kinds of path planning algorithms have been proposed, such as grid-based, potential fields, and sampling-based [20]. In this section the *grid-based* path planning algorithms are recalled, as they can be easily integrated with a terrain traversability assessment. Grid-based algorithms belong to the wider category of *graph search* algorithms as grid maps used in robotics can be actually seen as graphs, whose nodes are grid cells and edges between nodes are possible motions between such cells.

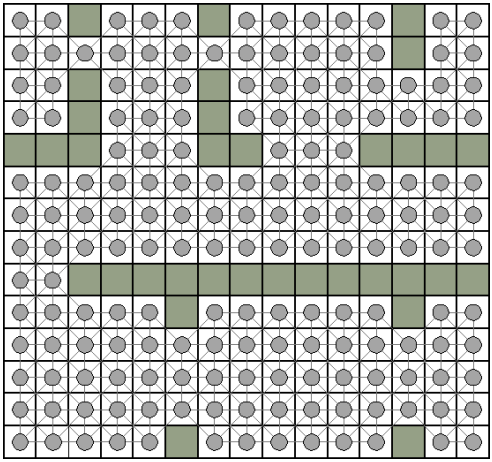
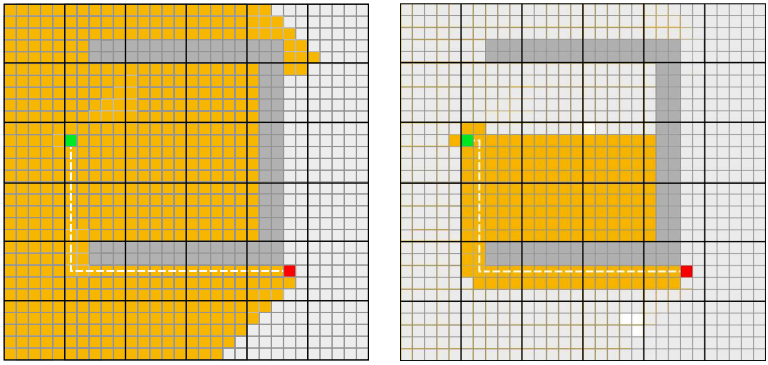


Figure 2.5 A grid map (obstacles in green) and the related graph (nodes and edges in gray)

One of the earliest graph search algorithms is the *Dijkstra's algorithm* [22]. It is an *uninformed search algorithm*, meaning that it is not driven to the goal in any way. However, it explores adjacent nodes, starting from the current node, according to the *arc cost* assigned to edges. These arc costs can be seen as the costs to reach the considered adjacent node. The algorithm prioritize paths with lower cumulative cost, while uniformly expanding until the goal is reached. Figure 2.6a reports an example of how this algorithm works in a rather simple 2D scenario (with a U-shape obstacle represented by the dark gray squares). The green square indicates the starting position, while the red one indicates the goal. The path found is indicated by the white dashed line and colored cells are those visited during the exploration to reach the goal. As it can be seen, Dijkstra's Algorithm finds a short path, but explores a rather large number of nodes.



(a) Dijkstra (b) A*

Figure 2.6 The Dijkstra's and A* algorithm compared in a 2D environment with obstacles

To overcome this issue the *A** algorithm was developed by Hart *et al.* in 1968 [37]. It still works with arc costs but, in addition, it uses a *heuristic function* to guide its search. For this reason the A* algorithm is referred to as an *informed search algorithm*. The heuristic function is usually an estimation of the distance to the goal: this allows the algorithm to give higher priority to those nodes closer to the goal, thus considerably reducing the number of explored nodes (Figure 2.6b). In particular the *cost function* of a generic node n is evaluated as

$$f(n) = g(n) + h(n) \tag{2.5}$$

where $g(n)$ represents the cost-so-far of the path from the starting node to current n node, derived by arc costs, and $h(n)$ represents the heuristic value, which is an *estimated path cost* from n to reach the goal. The idea behind A* is to evaluate the f cost of adjacent nodes for each iteration and to choose the one with minimum f in order to find the path with minimum cost. The A* algorithm can find the optimal path if the heuristic is *admissible*, i.e. if it never overestimates the actual cost from node n to the goal. An admissible heuristic is the *Euclidean distance*, which is "optimistic" as it neglects the presence of obstacles. It is, in fact, the most widely adopted heuristic function.

The A* algorithm is one of the most popular algorithms in robot planning because it allows to explicitly take into account the goal to be reached and, at the same time, it can divert from what would be the shortest path, in terms of Euclidean distance, depending on arc costs. Such costs are in the easiest case an estimation of the Euclidean distance from two adjacent nodes. However, they can

be derived by taking into account further aspects, such as altitude differences and, in general, the difficult of reaching a node. It is therefore understandable that such algorithm is particularly suitable for handling traversability costmaps. In fact, by properly relating cells cost to arc cost, the path planned by A* will be the shortest path (in terms of length covered) avoiding those cells with higher traversing costs. However, particular attention has to be paid to the order of magnitude of arc costs derived from the costmap, compared to the heuristic values. In fact, if the scales of the two values are too different, the resulting f cost will be mainly given by one of the two contributions, thus leading to a non-optimal working of the A* algorithm.

An improved version of A* is the *D** algorithm [87], whose name comes from *"Dynamic A*"*. In fact, the main enhancement of the D* algorithm, compared to A*, lies in the management of graphs with variable arc costs over time. In particular, whenever the graph changes, the A* algorithm has to recompute the whole path, while the D* performs an optimized *incremental replanning*, that allows to recompute only the portion of the path affected by the cost modification. This approach is crucial when the robot, while navigating to the goal, detects, from sensor readings, discrepancies between the original map and the actual situation.

2.4 Environment representation

In relation to the application considered, different types of models can be adopted to represent the environment where the vehicle has

to move in, with a different level of detail. One of the earliest 2D representations of the environment for robot navigation is the one using *binary occupancy grids* [24]. These are essentially grids (i.e. matrices) containing two values only, to discriminate the obstacle presence or not. A *non-defined* state can be also considered, when no information about the cell is available. The side of each grid cell has a physical size in real world, in accordance with the *spatial resolution* required. Despite the basic information provided, binary occupancy grids are still widely adopted as they can be easily recreated from laser-based mapping, which is a common approach for indoor navigation.

Focusing now on three-dimensional environment models, an intuitive and useful example is given by *Digital Elevation Models* (*DEMs*), also known as *elevation maps*. The environment to be represented is divided into cells from a top-view and for each cell the highest altitude is stored, thus obtaining a matrix of altitudes. The size of the cells determines again the spatial resolution, which is the distance, in the real world, between the elements of the DEM. The main drawback of the elevation maps consists in the impossibility to include overhanging structures or multiple height levels for one cell. For this reason, these models are typically referred to as 2.5D representations of the real world. However, they are an acceptable trade-off between computational burden and representation accuracy.

It is worth noting that both binary occupancy grids and digital elevation models are matrices. Therefore, they can be seen as *raster images* (respectively as binary and kind of gray-scale images).

This observation opens up to the possibility of employing tools that are normally applied to image processing to process maps as well. This approach has been adopted in different works described in the following chapters.

Finally, full-3D representations of the environment have been recently enhanced by the progress in *photogrammetry* and *photogrammetric reconstructions*. The latter belongs to the general methods of *structure-from-motion* [83], since it is a technique for 3D structures identification from sequences of images, if a suitable overlap is guaranteed. Furthermore, a *georeferenced* 3D model can be obtained if geographic coordinates are embedded in pictures acquisition. This point is particularly helpful in robotics, since georeferencing can be exploited for vehicle localization in a local reference frame of the environment. Nowadays, several mapping programs are available which can provide different kinds of 3D models, such as textured meshes or point clouds, by processing the acquired pictures.

2.5 Testing robotic platform: the U-Go robot

The robotic platform adopted to perform on-field tests of the works described in this thesis is *the U-Go Robot*, a tracked vehicle designed for rough terrain navigation [9], shown in Figure 2.7. It is 75 cm long and 88 cm large, with two 18 cm wide rubber tracks. It is localized with a Real-Time Kinematic Differential GPS, while its 3D attitude is acquired by a high-precision *Inertial Measurement Unit* (*IMU*), the MTi-30 by Xsense. The maximum speed the vehicle can reach is 0.8 m/s.

Figure 2.7 The U-Go robot, a tracked vehicle used in the experimental tests

The low-level control of the robot is managed by a Single-Board RIO by National Instruments including an FPGA for the real-time control of the two DC motors, programmed in the LabVIEW environment. It is responsible for the translation of velocity commands into motor velocities according to the kinematic model of the vehicle and the feedback signals coming from motor encoders. However, the locomotion principle of a tracked vehicle is rather similar to the one of a differential drive, as the two motors are used to drive the tracks in place of the wheels. Moreover, the FPGA acquires the sensor data such as IMU and GPS, as well as power information coming from the battery management system.

Higher-level functionalities of the robotic system are provided by a Raspberry Pi 3 board, running the *Robot Operating System* (*ROS*) [78]. ROS is a modular open source framework for robotics software development. Modularity is given by the so-called ROS

nodes, which are processes exchanging data one to each other and implementing specific tasks, such as motion planning, robot behavior monitoring, and data logging. This feature is extremely useful to test different control algorithms and, in general, ad-hoc nodes developed for experimental sessions. Furthermore, ROS sets up and manages communication between nodes through TCP-IP protocol, that allows to create a distributed computing system.

In the case of the U-Go robot, a wireless link is used to remotely monitor robot status, typically through a laptop. Data exchange between ROS and the Single-Board occurs via UDP connection and is managed by ROS nodes which collect raw data from the board, convert them into ROS messages and then publish such messages over the ROS network. Control commands can also be given to the U-Go platform via the ROS network as velocity commands, to drive the ground vehicle through the environment to a 2D goal pose, namely target position and orientation (see Sec. 2.1). In this manner, autonomous navigation strategies can be easily tested. Clearly, the platform can also be driven via common radio control. Figure 2.8 depicts the control architecture of the U-Go robot. Orange boxes are the main sensors and devices (computing boards in bold), while blue boxes are processes (related to ROS nodes). The dashed box encloses the ROS framework, while the wider solid red box delimits the part of the system onboard the vehicle.

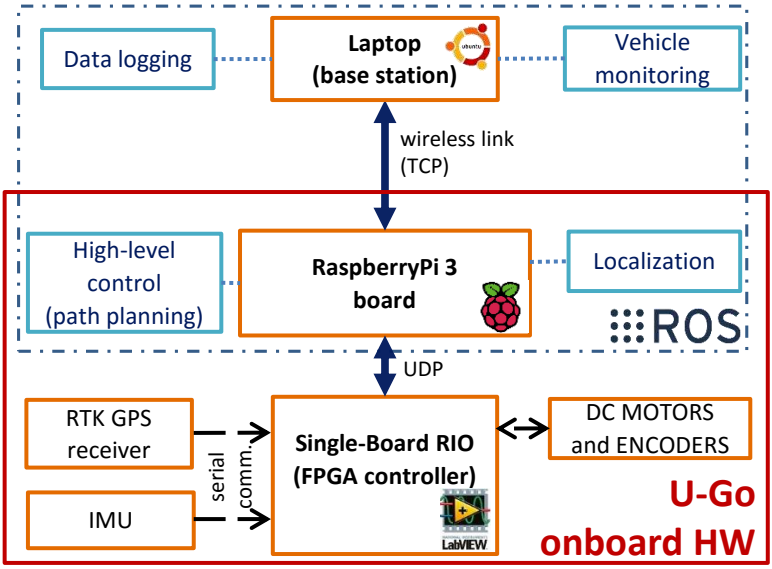


Figure 2.8 Control architecture and onboard hardware of the U-Go robot

Chapter 3

Geometry-based terrain traversability analysis

Whenever a UGV has to autonomously navigate in rough terrains or off-road areas, a proper path planning has to be adopted, taking into account the traversability capabilities of the considered robotic platform with respect to the environment to get through. As already mentioned in Sec. 1.2, the earliest solutions in this research field are based on a *geometrical analysis* of the environment, i.e. the terrain morphology or the shape of obstacles. This has been mainly carried out by processing lidar scans [3, 95], stereoscopic reconstructions [23, 86] or elevation maps of the environment [10, 46].

A growing number of recent works in the robotics literature relies on UAVs in order to provide an aerial overview of the considered environment or to perform drone-based mapping [6, 16, 32]. At the same time, photogrammetric 3D reconstruction from aerial surveys

has gained more and more relevance over the years, thanks to the enhanced quality of the results and the increased computation power available [21, 93]. Such kind of 3D modeling, can provide an accurate representation of the environment where the ground vehicle has to move, both from the geometrical perspective and from the appearance perspective, thanks to texture mapping. These features allow to perform a reliable terrain traversability assessment on the recreated environment model. Although this kind of processing on 3D models of the environment has been widely adopted in the literature [6, 11], only few examples exploit photogrammetry from airborne imagery to this end [27]. Another aspect is related to the assessment of rover navigation features. Some works apply learning models to infer vehicle's motion dexterity, while being remotely operated by an expert [73, 84].

In this chapter, a solution to the problem of the autonomous navigation of UGVs through unstructured environments is proposed, based on a geometry-based traversability analysis of the Digital Elevation Model of the considered region. The main novelty is the model derivation from a prior aerial photographic survey, carried out by an autonomous multi-rotor. Concerning vehicle navigation capabilities, no prior training stage is required, as no learning strategy is adopted. A set of prior experimental tests are, instead, proposed. After that, the traversability analysis is performed on the 3D model of the environment, by taking into account the estimated features of the rover. The final result of the processing is the map of costs related to the difficulty of crossing specific areas within the environment. This costmap is used with a graph-based path planning algorithm, in

order to “guide” the algorithm to plan a path minimizing the overall traversing cost.

3.1 Environment model reconstruction

One of the innovative aspects in this work is represented by the environment model reconstruction from *airborne imagery*. Initially, a multi-rotor autonomously performs a survey mission, in order to acquire regularly spaced pictures from above of the territory under inspection, by means of a stabilized on-board camera. In particular, during this mission the so-called *back-and-forth* or *boustrophedon* motion is performed by the vehicle, over a 2D polygonal area defined from the top-view of the considered region (Figure 3.1). All images acquired are geotagged with the GPS data of the UAV.

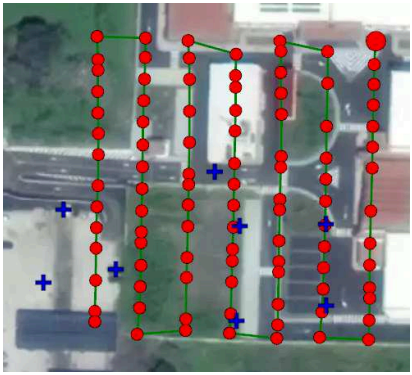


Figure 3.1 Example of back-and-forth motion for aerial imagery acquisition (pictures position represented as red circles)

After the acquisition of the pictures, the whole procedure for 3D photogrammetric reconstruction is carried out with the help of a professional software for drone-based mapping, Pix4Dmapper [76]. Among the several output formats, the software delivers the digital elevation model in UTM coordinates with ellipsoidal heights (WGS84). This is the representation of the environment used for the terrain traversability analysis, as explained in the following sections. Figure 3.2 shows the 3D reconstructions of two environments: a lava flow on Mt. Etna volcano (crossed by a trail) and the external area of the engineering department's laboratories of the University of Catania.

3.2 Vehicle navigation features

As already explained in Sec. 1.2, traversability is a property of the terrain closely related to the specific ground vehicle considered. This aspect can be intuitively understood by thinking of the different kinds of mobile robotic platforms available, in terms of locomotion, such as legged, wheeled, and tracked vehicles [83].

For every kind of vehicle, different parameters could be considered in order to define the motion capabilities of the platform, depending on their kinematic constraints. However, two indexes which can be evaluated for almost every type of ground mobile robots are the *maximum climbable height*, max_h, and the *maximum traversable slope*, max_{slope}. These are the two parameters used to derive the costmap, by performing two kinds of processing on the digital elevation model of the area considered.

(a) A lava flow on Mt. Etna volcano

(b) External area of engineering department's laboratories

Figure 3.2 Three-dimensional textured meshes reconstructed from the aerial survey

In particular, the georeferenced DEM is provided as input to a processing algorithm, along with the values of the two indexes, max_h and max_{slope}. Such algorithm delivers as an output two *binary occupancy grids* related to the two indexes. The algorithm consists of the following main steps:

1. surface normals estimation for each point of the DEM
2. height steps detection for each point of the DEM
3. binary occupancy grids generation from max_h and max_{slope} values

A detailed explanation of these steps is reported in the following sections.

3.3 Slope assessment

The *maximum traversable slope* allows to exclude from the environment those zones whose slope is higher than the allowed value. Furthermore, within the zone with a *traversable slope*, it is possible to distinguish those areas whose slope is closer to the maximum allowed from those almost horizontal (i.e. with a nearly null slope) by assigning high traversing costs to the former and low traversing costs to the latter.

A straightforward manner to perform the slope assessment is to consider *terrain normals*. The choice of using normals is justified by the possibility to determine, quantitatively, the slope exhibited by the terrain patch surrounding a given point of the DEM. Using

normals essentially the *z-component* of these vectors are considered. In fact, the value of this component equals the cosine of the slope of the plane tangent to the surface in the considered point, with respect to the horizontal plane of the model frame, which is an *ENU* frame, as shown in Figure 3.3. This approach exploiting terrain normals has been already adopted in the literature [81]. Furthermore, normals estimation is carried out on a slightly smoothed version of the DEM, since this choice has proved improving the quality of estimation, without modifying too much the original DEM.

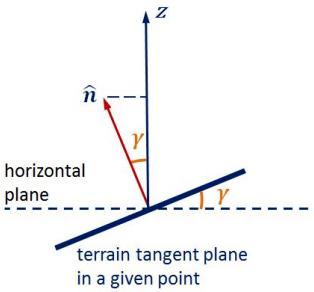


Figure 3.3 Normal z-component and tangent plane slope

Normals estimation is performed via *Principal Component Analysis* (*PCA*), by using the implementation employed in [91]. For each query point, the algorithm identifies a user-defined number of nearest neighbors. The best normal direction estimate of the tangent plane in the point is given by the eigenvector related to the *minimum eigenvalue* of the covariance matrix of distances between the neighbors and the centroid of the point set [80]. The eigen-based of estimation is a rather common approach in the traversability literature [39, 104]. In Figure 3.4 a graphical result is shown as a sample.

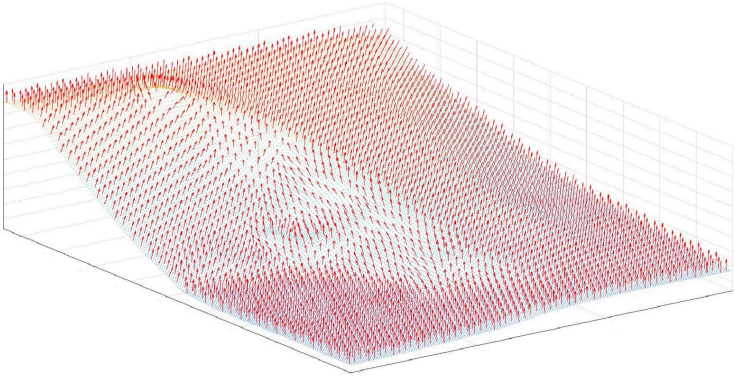


Figure 3.4 Detail of normals estimated over the terrain

The z-components of these normals are rearranged as a matrix, indicated as Z_{comp}, in order to have a one-to-one matching with the original DEM. Moreover, such components will be values ranging from 0 to 1, since unit vectors are being considered. Eventually, the matrix is compared with a threshold, which is the cosine of the maximum traversable slope; above this slope the terrain is marked as non-traversable. In particular, the thresholding is computed as

$$\text{slope grid}(i,j) = \begin{cases} \text{traversable} & \text{if } Z_{\text{comp}}(i,j) > \cos(\max_{\text{slope}}) \\ \text{non-traversable} & \text{if } Z_{\text{comp}}(i,j) \leq \cos(\max_{\text{slope}}) \end{cases} \tag{3.1}$$

where (i,j) is the element location within the grid. In this manner, the *first binary occupancy grid* is generated. Moreover, those areas all surrounded by non-traversable points (and then unreachable) are excluded by marking them as non-traversable as well. This step

has been implemented as a *region-filling* processing in the "binary image", i.e. the binary occupancy grid.

3.4 Height discontinuities detection

The *second binary occupancy* grid is derived as reported in the following. Whenever there is a step or a wall, there is a height discontinuity between adjacent cells in the DEM. Therefore a straightforward manner to detect this kind of obstacles is to visit each element of the DEM and to evaluate the absolute height differences with respect to its neighborhood (i.e. its eight adjacent surrounding elements in the matrix). If at least one of these differences is higher than the *maximum climbable height*, the considered cell is marked as non-traversable. Since such solution does not hold for the DEM boundaries, these are discarded in the following steps.

Eventually, a *binary segmentation* is performed on the occupancy grid. Unlike basic region filling described in previous section, this kind of image processing is typically used to identify objects boundaries by partitioning an image into sets of pixels. In this manner, the shapes of the physical obstacles (i.e. walls, cliffs, etc.) stand out, while unreachable isolated areas (i.e. all surrounded by non-traversable cells) are excluded, as in the previous case. In Figure 3.5 a sketch of all the processing steps carried out by the algorithm is reported.

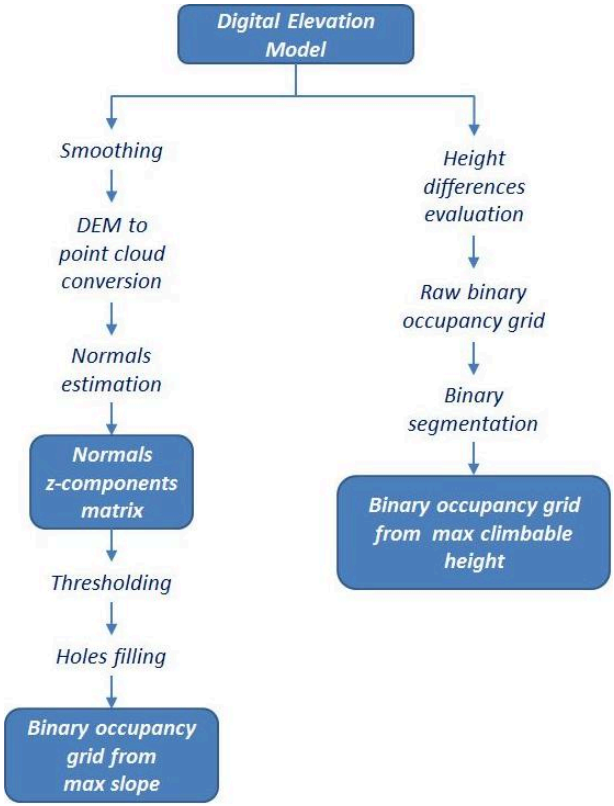


Figure 3.5 Scheme of the processing phases of the two occupancy grids generation

3.5 Traversability costmap derivation

The two occupancy grids described in the previous sections are only a background for the traversability costmap generation. A second algorithm is in charge of generating this costmap whose costs are, in general, *real values* greater than or equal to 1, unlike the *binary*. First, a *global* binary occupancy grid is built by merging the two occupancy grids, namely taking the union of their non-traversable cells. By indicating with one a non-traversable cell and with zero a traversable cell

$$\text{global grid(i,j)} = \begin{cases} 1 & \text{if slope grid}(i,j)=1 \vee \text{height disc. grid}(i,j)=1 \\ 0 & \text{elsewhere} \end{cases} \tag{3.2}$$

where (i, j) is the cell location within the grid. Clearly, if tight constraints are considered in deriving the two binary maps (namely a too low value for either max_h or max_{slope}), the global costmap could result chiefly covered by non-traversable areas thus making extremely hard to find a crossable path.

After grids merge, the traversing costs of *traversable* cells are derived as explained in the following. First, the costs of all cells too close to those marked as non-traversable are increased by dilating the non-traversable areas, by applying a Gaussian convolutional filter over the matrix. The standard deviation of the Gaussian filter establishes the kernel size of the convolutional filter and it has to be chosen by taking into account both the DEM spatial resolution and the vehicle footprint size. In particular, the kernel size, in meters, is related to the Gaussian standard deviation and the grid resolution (expressed in meters/cell) by the following

$$\text{kernel size} = (2 \cdot \lceil 2\sigma \rceil + 1) \cdot \textit{grid res} \tag{3.3}$$

which, in turn, determines the safety distance from non-traversable areas. The resulting matrix after dilation (which is no longer a binary matrix) is multiplied by a magnifying factor, w_{infl}. In this manner, only costs of the desired cells are increased, since the ones farther from obstacles still have a zero value after the dilation, which is obviously not affected by the multiplication.

A second contribution included in traversable cells' costs is the slope. In fact, although all cells with slope higher than max_{slope} are already marked as non-traversable, it is still preferable to avoid the robot traversing terrain patches with high slope and, in general, it is preferable to move along a path with as lower steepness as possible, both for safety and energy efficiency reasons. Therefore, starting from the matrix of normals z-components, Z_{comp} (described in Sec. 3.3), a higher cost is associated to those cells with inclination close to max_{slope} (i.e. whose values in the matrix approach $\cos(\text{max}_{\text{slope}})$), and lower costs are assigned to those cells with slope close to 0° (i.e. whose value in the matrix is close to $\cos(0°) = 1$).

Finally, it is possible to assign different weights to the two contributions. For instance it would be preferable for a wheeled or a tracked vehicle to favor paths with steps low enough to be climbed rather than steeper paths, while it could be the opposite for a legged vehicle. In particular the contribution of the inflated grid is weighted via the magnifying factor w_{infl}, while the costs derived from the slope assessment are weighted by another factor w_{slope}. Moreover, by acting on such magnifying factors, the ranging intervals for the values of the two contributions (i.e. slope and dilation) can be stretched or shrunk, thus allowing to give respectively a higher or lower relevance

to small variations of the considered contribution in the final cost. Such final cost is given by the following relationship, which holds for traversable cells only.

$$\text{costmap}(i,j) = w_{\text{infl}} \cdot \text{inflated costs}(i,j) + w_{\text{slope}} \cdot \text{slope costs}(i,j) \tag{3.4}$$

A further feature which could be easily included in the costmap derivation is the *terrain roughness*. In fact, PCA analysis allows to acquire information on both the terrain irregularity and the terrain normals at the same time, since they are respectively related to the smallest eigenvalue and the corresponding eigenvector of the covariance matrix of distances.

3.6 Experimental results

In this section the results of the described procedure are shown, carried out on the digital elevation models of the two different scenarios already presented in Sec. 3.1: the lava flow on Mt. Etna and the external area of the engineering department (Figure 3.2). With an average flight altitude of 30 m and an onboard camera of 12.4 MP, the related digital elevation models have a spatial resolution lower than 1 cm/cell (0.986 cm/cell). Furthermore, the results obtained by using the derived costmap with a graph-based path planner are presented.

3.6.1 DEM pre-processing

As a first pre-processing step, the DEMs are downsampled, thus obtaining a final *spatial resolution* of 9.86 cm/cell. This downsampling is performed by taking into account the size of the U-Go robotic platform, in view of the following experimental sessions. In particular, since the vehicle tracks are 18 cm wide, working with height values spaced every 10 cm is a fair approximation of the terrain surface over which the robot is supposed to move. Furthermore, this downsampling reduces the computational burden in the following processing steps.

3.6.2 U-Go traversability features

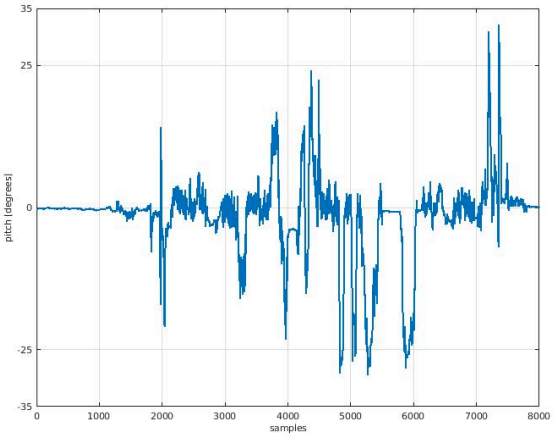


Figure 3.6 Sample chart of the pitch angle values

In order to establish both the max_{slope} and the max_h indexes for the U-Go robot, it has been teleoperated over some testing areas, bringing its navigation performances to the limit. During these first experimental sessions, proprioceptive data have been acquired; in particular the *robot pose* (namely roll, pitch and yaw angles and the GPS coordinates of the vehicle). Inspection of the pitch excursion has shown a minimum and a maximum value, recorded for this angle, of $-29°$ and $32°$ respectively. However, most of the values were within the $[-25°, 25°]$ interval, whose limits are around the 80% of the two extreme values detected. A sample chart of the acquired pitch values during a session is reported in Figure 3.6.

Therefore, the *maximum traversable slope*, max_{slope} has been fixed to $25°$, in order to prevent for the vehicle overturning, thus including a safety margin. The *maximum climbable height*, max_h , that the U-Go is capable to overpass, was observed to be 15 cm.

3.6.3 Resulting costmaps

At this point the 2D map including traversing costs can be derived. First, the Z_{comp} matrix values have been normalized and remapped from the $[\cos(25°), 1]$ interval into the $[1, 10]$ interval (see Sec. 3.5). It is worth recalling that this remapping involves an inversion in the interval trend, since those Z_{comp} values that were closer to $\cos(25°) \cong 0.906$ are remapped into 10 in the slope costs matrix, while Z_{comp} values equal to 1 are maintained equal.

Eventually, these values are combined with the inflated costs as per Eq. 3.4. Summarizing, for the U-Go vehicle the parameter values considered are reported in table 3.1. In this manner, both contri-

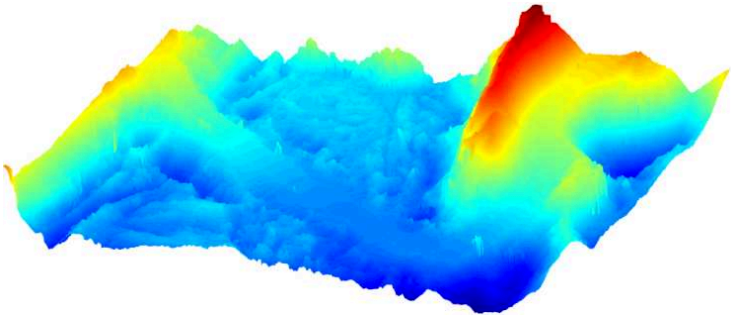
Table 3.1 Parameter values for costmap derivation

max_h	max_{slope}	w_{infl}	w_{slope}
0.15 m	25°	10	1

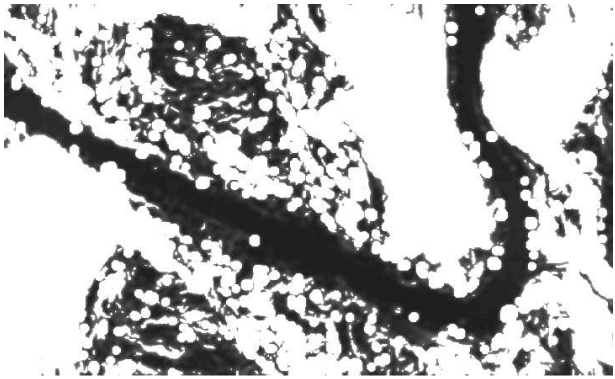
butions will fall into the [1,10] interval and, therefore, the resulting costmap will show traversing costs (except the non-traversable regions) in the [1,20] interval. With this choice, the two contributions will be treated equally in the path planning, which is the desired behavior for the tracked vehicle considered. Moreover, the intervals of the two individual costs proved to guarantee an acceptable sensitivity with respect to the cost variation, without having a too wide range.

In the case of the lava flow used in this work, starting from the digital elevation model (Figure 3.7a), the derived map of traversing costs is shown in 3.7b. White areas are non-traversable zones, while the black and gray zones are respectively easily traversable and hardly traversable areas. Thanks to slope estimation, the trail crossing the lava flow is clearly detected and mainly evaluated with low traversing costs. Within the traversable area, only a few regions are characterized by a slightly higher traversing cost (gray zones), while the remainder of the terrain is evaluated as non-traversable.

Considering now the external area of the departmental laboratories (DEM in Figure 3.8a), its resulting costmap is shown in Figure 3.8b. In this case a larger zone with high traversing cost can be seen (gray area in Figure 3.8b): this zone is the one having a slope not large enough to be marked as non-traversable, but which is still



(a) Digital elevation model (higher zones in red, lower zones in blue)



(b) Map of traversing costs in grayscale (non-traversable areas in white)

Figure 3.7 The lava flow on Mt. Etna volcano

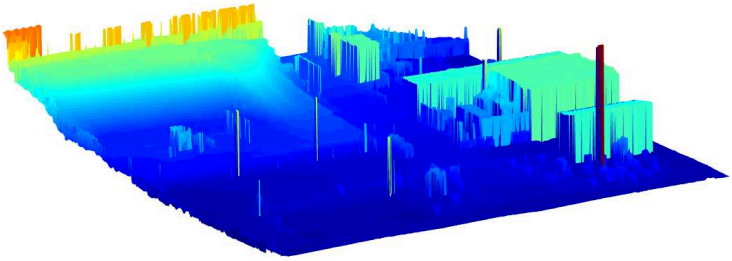
preferable to be avoided. Moreover, in this second scenario it is possible to verify that the discontinuity detection step works properly since lamp posts and buildings are correctly identified as obstacles (depicted in white in Figure 3.8b).

3.6.4 Global path planning results

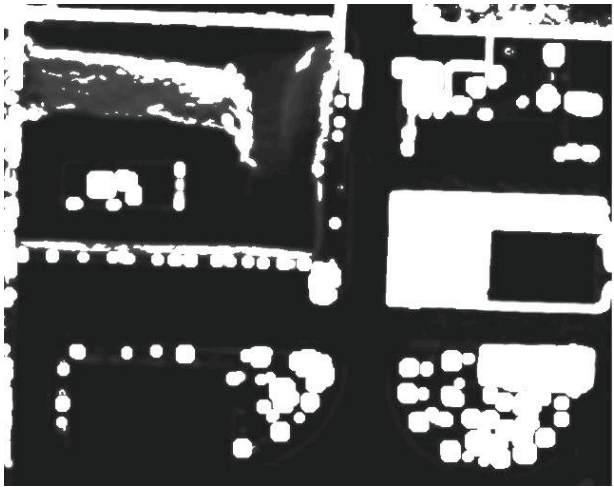
This section reports some preliminary results obtained by using the derived costmap with the D* graph-based path planning algorithm [87]. A specific subregion of the external area depicted in Figure 3.2b has been considered, which resembles a landslide scenario. Moreover, such subregion is interesting for testing, because it includes areas with different slopes and different degrees of navigation difficulty. The textured mesh of the region of interest is depicted in Figure 3.9a. The related digital elevation model is a 451×337 matrix, representing an area of 44.47×33.23 m depicted in Figure 3.9b.

In this case, in order to enhance the quality of the georeferencing derived by the mapping software, *Ground Control Points* (*GCP*s) have been included (Figure 3.10b). These are points within the scanned zone whose coordinates have been accurately measured via RTK surveying (Figure 3.10b) in order to be fed to the mapping software as "ground truth" in the geolocalization estimation. This procedure improves the georeferencing of the final model considerably, compared to the result obtained solely by images' geotags.

The D* algorithm allows to identify the *optimal path* in terms of traversing cost, taking into account the traversability costmap described in the previous section. The Euclidean distance has been considered as heuristic (see Sec. 2.3), while arc costs are computed



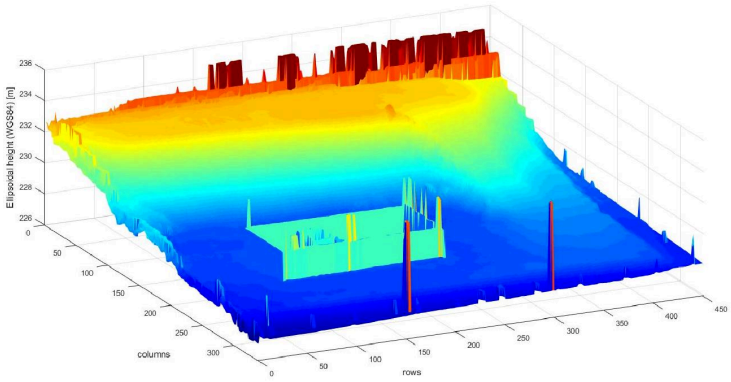
(a) Digital elevation model (higher zones in yellow, lower zones in blue)



(b) Map of traversing costs in grayscale (non-traversable areas in white)

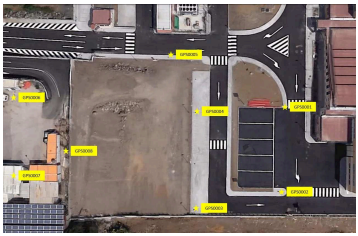
Figure 3.8 The external area of departmental laboratories

(a) Three-dimensional textured mesh



(b) Digital elevation model (higher zones in red, lower zones in blue)

Figure 3.9 Three-dimensional models of the landslide-like area



(a) Ground control points distribution in the area

(b) Ground control point surveying via RTK positioning

Figure 3.10 Ground control points distribution and their coordinate measurements

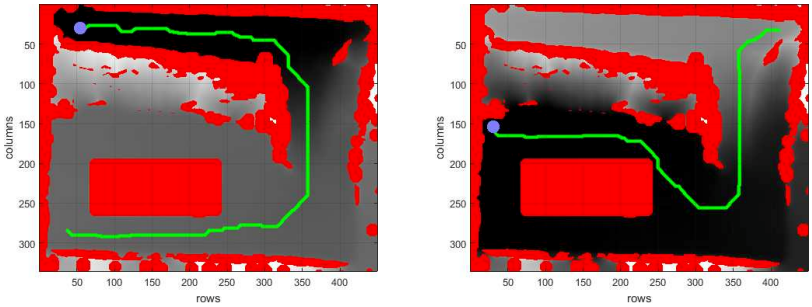
from cell costs as follows. In order to guarantee that the heuristic does not overestimate the cost to the goal, arc costs are given by the Euclidean distances between cells (by taking into account grid spatial resolution) as well, but weighted by a c factor. By indicating with δ the difference between the cost of the adjacent cell and the cost of the current cell, since costs range in the $[1,20]$ interval (Sec. 3.6.3), δ will range in the $[-19,19]$ interval. By shifting this range by 20, the new values will lie in the $[1,39]$ interval which is normalized, and then remapped into the $[1,10]$ interval. The derived value is such c factor. The idea in such definition of arc costs is essentially to favour to motion towards areas with a traversing cost lower than the cost of the current cell. In fact, a cell with a cost much larger than the one of the current cell will be "seen" almost ten times *farther* than its real distance. On the other hand, a cell with a cost much lower than the one of the current cell will be seen at its real distance. This procedure can be easily generalized, regardless of the specific values of the costmap, as reported in Algorithm 1.

Algorithm 1 Arc costs derivation from traversing costs

for all cells in *costmap* **do**
  for all cells surrounding *current cell* **do**
    δ = costmap(adj. cell) − costmap(curr. cell)
    $\delta = \delta + \max(\text{costmap})$ ▷ δ range shift in the $[1, 2 \cdot \max(\text{costmap})]$
    Normalize δ in the $[0, 1]$ interval
    $c = d_{\max} \cdot \delta$
▷ $d_{\max}$ determines the max magnifying effect on the Euclidean distance
  end for
end for

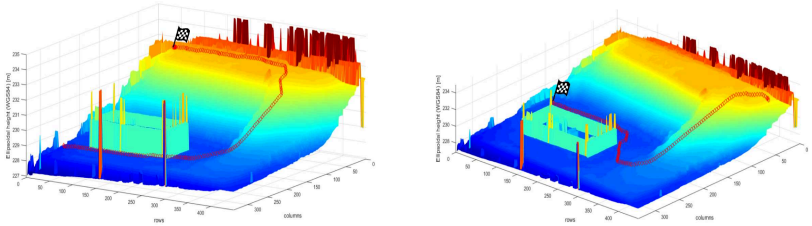
Two sample paths from two different couples of starting and final points are here reported. In Figure 3.11 these paths are shown: both the starting points and the final points (indicated with blue dots) are chosen to make the vehicle reaching the upper part of the area starting from the lower part and vice-versa. The lighter gray shades are those cells of the costmap characterized by a higher cost, while the red areas are non-traversable zones (i.e. those zones marked as obstacles in the global binary occupancy grid described in Sec. 3.5). It is worth noting that, although in the real world no physical obstacle is present, the path planner identifies a path to reach the final target which is not the shortest, in terms of Euclidean distance in the real world, but it rather is the *shortest* in terms of the traversing cost. This is due to the fact that the prior traversability analysis detected a wide area as absolutely non-traversable.

Afterwards, these trajectories are converted into a sequence of waypoints in UTM coordinates. In turn, a ROS node converts such UTM coordinates into local coordinates, with the respect to the local map frame of the environment, and eventually sends them to the



(a) path #1 (path length: 75.62 m) (b) path #2 (path length: 56.20 m)

Figure 3.11 Planned paths by the D* algorithm based on the derived costmap



(a) path #1 (b) path #2

Figure 3.12 Track of the vehicle over the terrain

navigation controller of the tracked vehicle. The node sends the following point as soon as the previous one has been reached. In Figure 3.12, the red tracks represent the 3D reconstructions of the paths over the terrain surface.

3.7 Conclusions

In this chapter a strategy for unmanned ground vehicles navigation in unstructured environments, based on a previous autonomous survey of the environment performed by a UAV, has been presented. A geometry-based traversability analysis of the terrain model produces a final map including the costs of crossing the different areas of the considered environment. By providing this map to a path planning algorithm it is possible to avoid the crossing of zones with high traversing cost, which also prevents the vehicle to get through dangerous areas, where it will be likely for the vehicle to overturn. Furthermore, a longer autonomy in performing the mission is obtained by preferring less steep paths, thus allowing a reduced power consumption while the platform is moving. An example of usage of such costmap for a tracked vehicle with a D* planner has also been presented. The achieved results in terms of autonomous navigation capability of the rover are promising. A further investigation will be a systematic comparison with other methods available in literature.

Although the main idea of the overall work is a recurring method in the related literature, the adoption of aerial photogrammetric reconstruction in this research field is the main innovation. Thanks to the quite simple kind of analysis performed on the surface model, the procedure allows to obtain quickly a traversable path, avoiding dangerous steep slopes and steps. Moreover, this solution can constitute a backbone to be integrated within a local obstacle avoidance controller, in order to support dynamic changes of the map, with the help of the optimized replanning method of the D* algorithm.

The information on the terrain roughness was not included in the final costmap presented in this chapter since, in the considered case study, it proved to be only a cause of noise in the planned path, without any remarkable increase in quality. This was due to the fact that strongly irregular areas were already marked as obstacles. This outcome would suggest to take into account roughness information in the local path planning rather than in the global one.

Chapter 4

Robotics teleoperation in outdoor environments

In the context of outdoor environments, there are some scenarios which are still too challenging to be faced with a fully autonomous approach, based on terrain traversability analysis. In these situations the solution mainly adopted in robotics research is *teleoperation*. In robotics teleoperation, the *vehicle-operator interaction* occurs by means of an *interface*, due to the lack of direct visual contact with the vehicle, time delay, or large physical distances [29].

In this chapter a novel interface for the problem of remote control of ground robots is presented. The proposed interface exploits *Mixed Reality* (*MR*) to make more intuitive visual communication of exteroceptive and proprioceptive data. The use of so-called *visual aids* (*VA*) is discussed, designed coherently to the information type to be displayed. Such data communication approach enhances *situational*

awareness and facilitates *decision-making* of the teleoperator. These aspects are related to the comprehension of the surrounding environment, of the robot location within that and of the robot-environment interaction [25, 52, 103]. Moreover, experimentations are reported in order to verify feasibility of the proposed method on a real robotic system operating outdoor.

4.1 Teleoperation interfaces

A teleoperation interface unavoidably affects the way data and environment are displayed and, therefore, how they are perceived by the operator. A properly designed teleoperation interface would make the driver capable of understanding the operating remote environment, allowing to take actions and provide commands. At the same time, a trade-off between the amount of information transferred and the cognitive-sensorimotor workload minimization needs to be found [29]. A properly designed interface is therefore the one making human-machine interaction smooth and intuitive.

Telerobotics concerns a wide spectrum of robotic platforms and applications. Over the last decades, teleoperation interfaces based on mixed reality have experienced use on a wide range of applications, including robotic tele-manipulation [19, 53, 102], and telesurgery [69, 77]. In particular, *Virtual Reality* (*VR*) allows to recreate external perspectives, which can dramatically simplify a driving task [29]. VR has also been applied to interfaces for underwater robots [48, 59], and aerial vehicles [13, 71].

However, few works in the literature have addressed the problem of remotely driving an unmanned ground vehicle leveraging virtual reality and visual aids, especially in the outdoors. In the following subsections a review of current state of the art in teleoperation interfaces is given. Works are sorted in relation to their testing environment, namely indoor and outdoor scenarios.

4.1.1 Indoor solutions

Nielsen *et al.* [63] provide a remarkable example among the solutions for indoor. The proposed interface is formally based on the concept of *affordances* theorized by Gibson, according to which an environment inherently includes all the information to interact properly with it [31]. The authors apply this theory to improve robot's navigation skills by combining different kinds of information (namely 3D environment map, robot model, position and orientation), within an *Augmented Virtuality* (*AV*) context where the video from the onboard camera is projected on a rectangular screen at a fixed distance in front of the robot, thus obtaining a single integrated view (Figure 4.1). Authors put particular emphasis on fulfilling three design principles: data presentation in a common reference frame, which relieves the operator from spatially relating the different pieces of information, correlation between expected response and given commands, and adjustable perspective. Commands to the platform are issued through a joystick, which is also used to move the onboard pan-tilt-zoom camera. The map of the operating area is prior known or reconstructed from laser or sonar sensors. The paper reports experiments carried out both in simulation and in real world. The

proposed interface is compared against conventional 2D interfaces, where data and information are spread in separate parts of the display instead. Test results proved the effectiveness of the developed interface, even with 1 s of delay between video streaming and control commands transmission. Despite being 3D, the user interface by Nielsen *et al.* lacks of stereopsis generation, since it is displayed using a standard 2D screen and the video is acquired by a single camera.

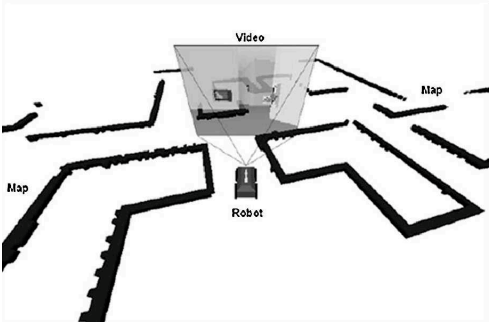


Figure 4.1 Single integrated view of the interface by Nielsen *et al.* (image from [63] ©2007, IEEE)

Ferland *et al.* [28] developed a control station screen, including textual reports on robot status in the lower left corner (battery and odometry), and camera images in a small window in the upper left corner. In the center of the screen, besides the robot graphical model, a 3D video projection is performed based on either laser scans or stereoscopy (Figure 4.2). With the laser-based projection, a mesh of floor and walls is built. Such mesh covers the whole camera field of view and its vertexes are projected on the camera left image, through the application of the equation of perspective projection. The sec-

ond projection approach exploits stereoscopic disparity to estimate depth and to derive the surface on which camera images are mapped as a texture. Also in this work, adjustable viewpoint is considered, along with five different pre-set positions, including both egocentric and exocentric perspectives. The exocentric camera heads toward a target point fixed in front of the robot and the user can change the perspective with respect to such point. Furthermore, a square grid, current laser range data and concentric circles (centered on the robot model) are included as further visual aids. The vehicle is equipped with a stereo camera and a 2D laser scanner, required to perform the two projection approaches, whilst a wireless joypad for driving the platform is used. A global 2D map of the environment is reconstructed and continuously updated with incoming laser scans. To assess the interface usability, authors carried out a pilot study, with the help of an eye-tracker to establish the usefulness of the different elements of the interface. Although the hardware equipment includes a stereo camera, it is solely used for disparity estimation in the stereo-projection approach. Moreover, such approach often proved to be unstable and highly distorted during testing sessions. The interface is not displayed through a stereo screen, thus losing the depth perception when shown to the operator.

Similarly to [28], Livatino *et al.* [49] exploit *Augmented Reality* (*AR*) visualization, to display both video and laser range measurements in a spatially and contextually coherent fashion. In particular, colored overlays representing sensor information are integrated in the incoming video images 4.3. In order to do that, a crucial aspect is the graphical alignment of data representation. This is done via

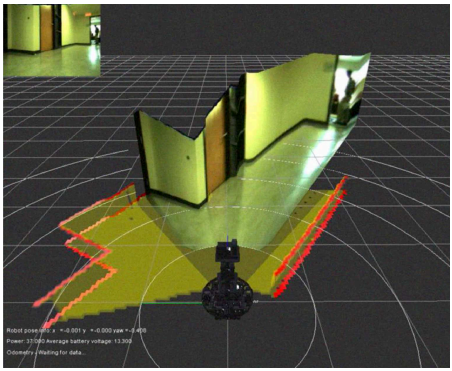


Figure 4.2 A view of the interface developed by Ferland *et al.* (image from [28] ©2009, IEEE)

a brief calibration process, during which the operator is guided in matching virtual elements with real images. The shape and the initial 3D position of every visual aid are derived from a computer vision algorithm applied on the scenario pictures. The interface is presented to operators with stereoscopic viewing, including both real scene, captured by the onboard stereo camera, and virtual objects, namely semi-transparent colored walls or colored lines (representing laser rays), where color-coding is used to represent the detected distance. Further optional virtual elements are the numerical values of measured distances, shown as plain text, and a circular grid centered on the robot. Tests have been performed in indoor factory-like environments, with users driving the mobile robot using a desktop computer keyboard, located several kilometers apart from the operator control station. During tests, the observed communication delay was, on average, 1 s and the proposed interface was still positively judged

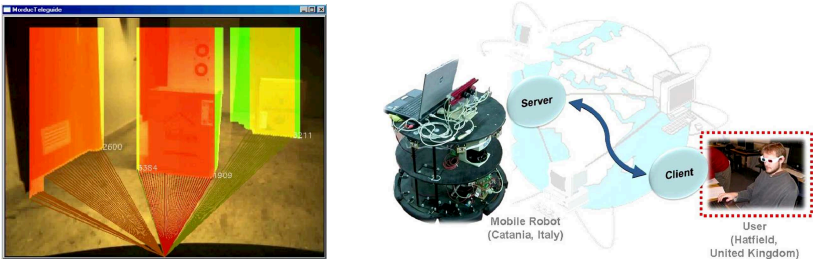


Figure 4.3 A view of the interface developed by Livatino *et al.* and a scheme of the teleoperation system (images from [49] ©2012, IEEE)

by the operators. However, as reported by authors, the approach is suitable for quite simple indoor environments, since the processing algorithm adopted needs to identify edges to define the ground floor. Furthermore, no head-mounted display can be used since the point of view is bound to the one of the real onboard camera. Therefore, the stereoscopic visualization cannot be "immersive".

The paper of Okura *et al.* [70] focuses on the problem of the viewpoint in robotics teleoperation. Authors exploit computer vision to overcome the difficulty of capturing a third-person view from a real camera. In detail, by means of an omni-directional camera they recreate free viewpoints by reconstructing a rendering surface from the stitching of the point clouds captured by four Microsoft Kinects. The rectified images from the omni-directional camera are finally mapped over such surface, thus obtaining photo-realistic textures.

The 3D reconstruction of the environment is enhanced with a graphical model of the robot and an arrow (displaying robot heading), both as semi-transparent virtual objects. Driving commands are issued via a common gamepad and the teleoperation interface is

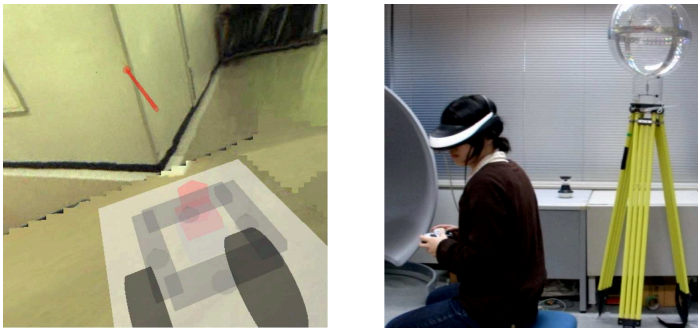


Figure 4.4 A view of the interface developed by Okura *et al.* and the operator interface with HMD (images from [70] ©2013, IEEE)

shown to the operator with an *Head-Mounted Display* (*HMD*) (Figure 4.4). The proposed interface has been tested in both virtual and real indoor environments, with three perspectives: first-person view, third-person view, and free viewpoint. During the trials in the real world, a 3D reconstruction of the environment was concurrently carried out, by using odometry information and by merging the acquired point clouds over time. Authors state that the free-viewpoint effectively enhanced the accuracy of operators' maneuvers. The main limit of this work is the use of Kinects, which make the proposed solution unsuitable for daylight outdoor scenarios. Furthermore, because of the small-size and semi-transparency, the introduced visual aids appear to be often poorly visible during the vehicle navigation.

Zalud [105] deals with the reconnaissance of hazardous areas for the search of victims. The operator wears a head-mounted display, including head tracking, which is used to create "Virtual Head-Up Displays". These are small displays showing data coming from

onboard sensors (namely a color camera, a thermal camera and a nodding laser scanner) along the sides of real camera images, in an augmented reality context (Figure 4.5). Moreover, the interface provides various "data fusion visualization" modalities. For instance, data of the thermal camera can be displayed as a colored layer on main camera images, while robot attitude is indicated by means of a semi-transparent artificial horizon. Distances acquired by 3D laser scanner can also be included, through color-coding. Commands are issued to the platform via a two-hand airplane-like joystick. This driving concept is typically referred to as *Hands on Throttle and Stick*, which has the advantage to prevent the operator from using different devices. Laser scans are used to build a 2D occupancy grid map of the environment, thanks to a SLAM algorithm.

Calibration of matching between 3D laser data and RGB image is pointed out as one of the most delicate problems. It is carried out to project 3D data onto the image and, vice-versa, to get the color information by inverse projection. Another issue faced is the spatial correspondence between RGB and thermal camera images, due to their different fields of view and resolutions. Despite the augmented reality visualization, the presented solution suffers for lacking of depth perception, since a monocular color camera is used, which limits the potential of the HMD. Furthermore, the virtual head-up displays show only a partial information (like 2D and 3D scans, 2D map, robot status and text messages) in separate parts of the operator viewed frame. This could result in a cognitive overload for the user.

Martins and Ventura [55] designed an interface for *Search And Rescue* (*SAR*) applications. They proposed a head-mounted display

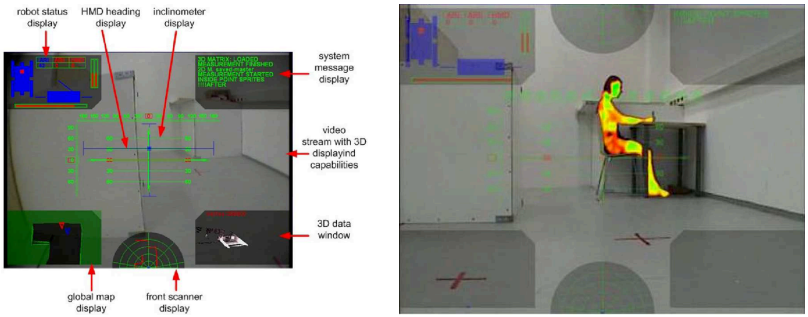


Figure 4.5 Examples of data fusion visualization in the interface developed by Zalud (images from [105] ©2006, IEEE)

to show the video stream from a stereo camera onboard the vehicle. The head tracker embedded in the HMD is used to directly control yaw and pitch of their SAR robotic platform. Rotations along the roll axis detected by the head tracker are used to rotate accordingly video frames by image processing. Forward and backward motion are controlled via a gamepad (Figure 4.6). Thanks to the stereoscopic acquisition, operators experienced a high spatial awareness and depth perception during experimental tests. However, the heading control via HMD made the interface counterintuitive for most of the users and actually worsened its usability. Finally, the video is not integrated with any sensor data, as this enhancement was left for future developments.

Krückel *et al.* [45] proposed what they call a "free-look" interface thanks to a spherical camera, which is in principle very close to the one described in [70]. The images acquired and rectified from the spherical camera are displayed, through an HMD, integrated with an artificial horizon representing robot attitude (Figure 4.7). Further-

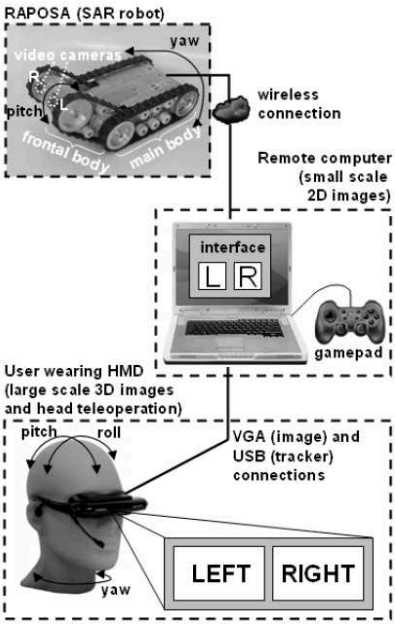


Figure 4.6 The scheme of the teleoperation system proposed by Martins and Ventura (image from [55] ©2009, IEEE)

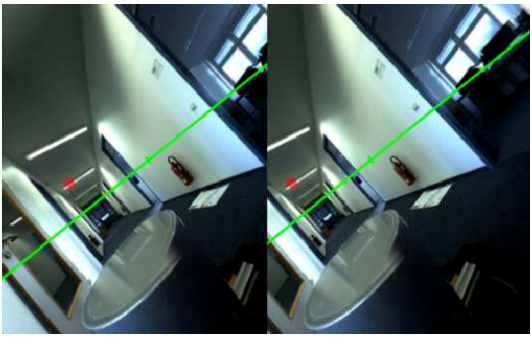


Figure 4.7 Example of views of the interface developed by Krückel *et al.* (image from [45] ©2015, IEEE)

more, multi-user sessions are possible, allowing several operators, each of them equipped with HMD, to see the same scene simultaneously. Robot control is left to one operator only, while other users are observers that can help the main operator with their different perspectives. The viewports of other users' cameras are also added as visual aids.

The interface is enhanced with a basic force-feedback, which is given by vibrations of the game controller used to drive the robotic platform. Tests have been performed on a four-wheel UGV with limited rough-terrain navigation capabilities. However, only indoor environments were considered. Moreover, no information about the 3D geometry of such environments is taken into account during robot navigation. Finally, although the interface is presented via HMD, camera acquisition is not stereoscopic.

4.1.2 Outdoor solutions

Planetary exploration is a robotics research area inherently related to teleoperation in outdoor, as reported by Nguyen *et al.* [61]. This paper is a review of all previous works about interfaces for planetary rovers remote driving, leveraging virtual reality. When dealing with such application scenarios, a number of issues arise, especially in the telecommunication, such as large delays, intermittency, drops, and limited bandwidth. Hence, the need to develop robust interfaces to remotely control the platforms.

In particular, the paper presents the progressive improvements of the quality in interface design, starting from the first *Virtual Environment Vehicle Interface* [75], exploiting virtual reality and 3D

visualization for intuitive robot status displaying. Successive works underline the relevance of freely changing perspective around the robot, to enhance "telepresence" [99] and the use of stereoscopy [88]. The last interface described is named *Viz*. It provides tools for the interaction with the virtual environment, such as information query, terrain model analysis and objects handling. An interesting feature is the possibility to virtually introduce well-known objects in the scene, in order to have an intuitive term of comparison for the user (e.g. a human silhouette or a soda can).

A modern version of teleoperation interface for planetary exploration is *VERVE* (*Visual Environment for Remote Virtual Exploration*) [96], with highly enhanced 3D graphics quality (Figure 4.8). It includes more visual aids than earlier interfaces did (e.g. directional indicators, compass, etc.) and thanks to its modularity it can be integrated with further functionalities. However, a feature not fully considered in the above works is the graphical visualization of geometry-related information of the planetary terrain. Moreover, although head-mounted displays have been used for decades in astronauts training, they have not been particularly exploited as visual interfaces for rover teleoperation.

Still in the outdoor context, Miller and Machulis [58] focus on designing proper visual aids to support the operator while controlling a UGV for planetary exploration. In particular, the proposed interface includes an artificial horizon, a few icons indicating robot platform condition (i.e. communication link, camera, control and GPS statuses) and a semi-transparent minimap, reporting the covered path (Figure 4.9). Along with them, authors introduce a novel visual aid,

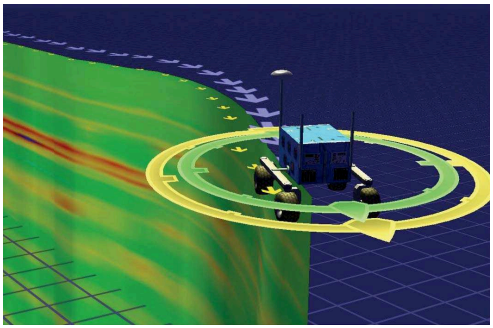


Figure 4.8 Visual Environment for Remote Virtual Exploration (*VERVE*) (image from [96])

which helps the operator in issuing commands in presence of delay. It consists of a cross representing current control command (sent with a joystick or a keyboard), and the trace of previous commands over time is graphically kept, depending on the current amount of delay in the control command communication.

The robotic platform has two cameras, providing the couple of stereo images, which are presented to the operator in combination with the proposed graphical elements. The interface is conceived for an immersive head-mounted display; however, tests have been performed with a stereo display, both in simulation and in real lunar-like environments, with very demanding conditions for the operator (time-delays up to 4 seconds and less than 2 fps of low jpeg quality camera images). The operators confirmed the helpfulness of the command visual indicator, in managing the command-response delay. However, all the visual aids proposed in [58] are related to robot status, while no information about the surrounding environment is

included in the operator visual frame (with the only exception of the minimap).

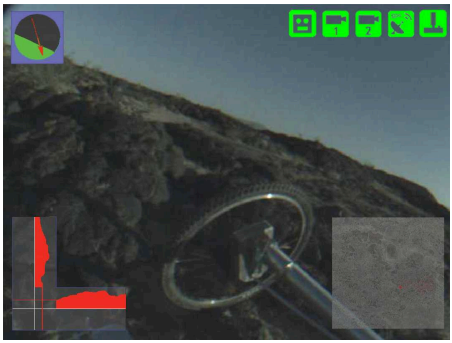


Figure 4.9 A view of the interface by Miller and Machulis for lunar exploration (image from [58] ©2005, European Space Agency)

An interesting work is the one described by Kelly *et al.* [41]. In their paper, the authors present a solution that leverages real-time 3D reconstruction of the environment with the help of what they call a photogeometric sensor. From the acquired data, a "virtualized reality" is built, allowing to generate useful viewpoints for the vehicle teleoperation. The photogeometric sensor is composed of a stereo camera, an infrared camera and a nodding laser scanner, thus providing visual appearance along with per-pixel depth information. Furthermore, the photorealistic environment reconstruction is enriched by introducing the 3D robot model, a virtual speed indicator, and the covered path trace (Figure 4.10) .

Another kind of augmentation is also proposed, such as representing terrain slope via color-coding. In particular, they propose to mix real color with red hues for those areas to be avoided, and with

green hues for those zones that are easily navigable. They advise this visualization mode could be effective in extremely dark conditions, thanks to both LIDAR and infrared camera. The operator control station is made up of a large monitor, while commands are supplied through a steering wheel and pedals. The visual interface provides three perspectives: raw video, over-the-shoulder and birds-eye view. To tackle the delay in commands transmission, a latency compensation approach based on a "virtual leader follower" is used. Thanks to this strategy, during the usability assessments carried out through on-field tests, remote operators were capable to drive at 24 km/h, even on very rough terrains. The main drawback of the proposed solution is the large number and cost of onboard sensing devices. Moreover, since the interface is shown on a common 2D screen, the operator cannot perceive the depth information included in the real-time 3D reconstruction, thus limiting the potential of the proposed method.

Figure 4.10 The operator control station and a view of the interface by Kelly *et al.* (image from [41] ©2011, SAGE Publications)

Stückler *et al.* [89] developed a semi-autonomous mobile manipulator for the 2013 DLR SpaceBot Cup. The designed control interface features three levels of autonomy: full autonomy, when the

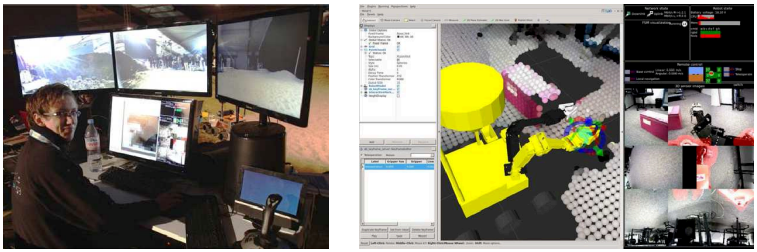


Figure 4.11 The teleoperator station and the graphical interface developed for the 2013 DLR SpaceBot Cup by Stückler *et al.* (images from [89] ©2015, John Wiley and Sons)

operator assigns the entire mission, semi-autonomy, when the robot executes a sub-task on operator request (e.g. move to a goal, grab an object), and, finally, direct teleoperation, when the platform and the arm are driven through a joystick. The control station consists of three monitors, showing the real images coming from three frontal webcams, while a fourth monitor includes telemetry information (e.g. robot attitude acquired from IMU) and controls for teleoperation as well (Figure 4.11). An overhead wide-angle camera is also included, to have a quick overview of the robot surroundings. The mobile platform is a six-wheel skid-steering robot with a 7-DOF arm. Geometry of the environment is acquired from eight RGB-D cameras and a rotating 2D laser scanner. Such sensing data are used both for SLAM, allowing to change the viewpoint within the reconstructed virtual environment, and obstacle avoidance.

The direct teleoperation interface is tested by a skilled operator, focusing on the telemanipulation task, with a 2 s delay communication link, in order to simulate the condition of the challenge. It is pos-

sible to include a downsampled version of the acquired point-clouds of the surroundings and a "phantom" of the arm position according to the planned path, which can be consequently re-planned. However, no support is provided to the user when remotely driving the robot. The traversability assessment performed on the acquired 3D data is used for path planning only, while such information could be graphically shown to the operator. Furthermore, a large number of sensors is used for 3D reconstruction, but the user interface is displayed on 2D monitors only. For this reason, as a future development, authors claim they want to improve the situational awareness by introducing HMD or 3D display. In a later work [82], a head-mounted display is proposed to enhance telemanipulation only.

Tables 4.1 and 4.2 provide a summary of the works described in this section and their proposed visual aids.

4.2 Proposed interface

The interfaces described in the previous section present very different features and offer various elements to support the operator while remotely driving a mobile robotic platform. Some of them exploit mixed reality to include information on robot status, environment and sensing data as visual aids. Most of the times, these virtual elements display exclusively proprioceptive sensors data. Only in [41] some information about the remote terrain slope is given to the teleoperator. Other interfaces recommend the use of stereo cameras to increase spatial awareness; however, only few of them [55, 70] combine stereopsis with the use of immersive displays (such as HMDs).

Table 4.1 Summary of the proposed visual aids

	Testing scenario	**Onboard camera**	**Command tools**
2005-Miller and Machulis [58]	outdoor	stereo and mono	keyboard and joystick
2006-Zalud [105]	indoor	mono	two-hand joystick
2007-Nielsen *et al.* [63]	indoor	mono	joystick
2009-Martins and Ventura [55]	indoor	stereo	gamepad and HMD head tracker
2009-Ferland *et al.* [28]	indoor	stereo	gamepad
2011-Kelly *et al.* [41]	outdoor	stereo and infrared	steering wheel and foot pedals
2012-Livatino *et al.* [49]	indoor	stereo	keyboard
2013-Okura *et al.* [70]	indoor	spherical	gamepad
2015-Krückel *et al.* [45]	indoor	spherical	gamepad
2016-Stückler *et al.* [89]	outdoor	3 mono and 1 wide-angle	joystick

Table 4.2 Summary of teleoperation interfaces described

	Visual aids
2005-Miller and Machulis [58]	artificial horizon, robot status icons, minimap, command visualizer
2006-Zalud [105]	artificial horizon, virtual head-up displays, minimap, data fusion visualization
2007-Nielsen *et al.* [63]	adjustable perspective, 3D robot model, 3D map
2009-Martins and Ventura [55]	none
2009-Ferland *et al.* [28]	adjustable perspective, square grid, ground concentric circles, 3D robot model, laser scans, single-camera mini display, real-time video mapping
2011-Kelly *et al.* [41]	adjustable perspective, speed indicator, 3D vehicle model
2012-Livatino *et al.* [49]	proximity walls, laser rays, ground concentric circles, distance numerical values
2013-Okura *et al.* [70]	adjustable perspective, 3D robot model, heading arrow
2015-Krückel *et al.* [45]	adjustable perspective, artificial horizon and heading
2016-Stückler *et al.* [89]	adjustable perspective, 3D robot model, downsampled point-cloud, phantom arm position

Another crucial point is the lack of use of a 3D model of the region where robot is navigating. Most of the works reported in the previous sections performs SLAM or, more generally, real-time 3D reconstruction. Others do not take into account any information about the environment geometry at all. If such information is available, prior to the teleoperation session, the operator interface can be enhanced by including further instructions and indications, derived from the environment geometry.

The novel interface proposed in this chapter combines the most helpful aspects described in others' work. A stereo camera is used to acquire images from the real remote environment. The acquired frames are combined with a 3D reconstruction of the area of interest, in a virtual context. The advantage of using this approach is twofold. First, it allows to derive synthetic perspectives, preserving the spatial relationships between all the elements. This increases operator situational awareness. Secondly, the virtual environment can be easily enriched with visual aids for the driving task. Particular emphasis has been put on operator "awareness" while planning a path in the remote environment, with visual aids guiding the driver to avoid difficult terrain areas. This information can be extremely useful for UGV teleoperation, especially in outdoor unstructured scenarios. Finally, the visual interface is presented by means of a head-mounted display, thus guaranteeing the most immersive experience for the driver.

4.2.1 Interface features

According to [56], the MR interface developed falls into the *augmented virtuality* context, which embeds visual aids to support teleoperation. Both real and virtual elements are integrated three-dimensionally on operator's display. The main features of the designed interface are:

- single-frame integrated view
- immersive visualization
- visual aids

Single-frame integrated view

All the information, useful to accomplish the teleoperation task, is presented to the operator on a single screen. This solution was preferred rather than separating the information over different screens and presenting them at the same time, as done for instance in [58, 63, 89, 105]. In fact, concurrent visualization can lead to cognitive overload, raising the problem of *divided attention*.

Moreover, a *three-dimensional* mixed reality environment is considered in order to display all the information available in a single integrated view, by combining live video stream, graphical representation of the environment and all the other visual aids. In particular, the real scenario is represented through streamed video frames, since these are deemed as the most intuitive and natural way to show current conditions of both the environment and the robot. For this

reason, to further enhance the perception of remote area, a *stereoscopic* video stream is considered and shown as stereoscopic 3D view within the single viewing frame.

Clearly, three-dimensionality of the interface facilitates users' depth comprehension, especially in presenting 3D visual aids. However, such approach raises the issue of spatial alignment of all the graphic elements in the 3D viewing port. In particular, the two stereoscopic 3D views (video from the real world and user view of the virtual environment) need to be consistently aligned.

To solve this problem, a *calibration tool* is proposed in the current interface implementation, allowing the operator to manually adjust the perspective. This is crucial especially to fix possible mismatches between video frames and the reconstructed environment outline, which would potentially reduce effectiveness of the 3D visual aids and cause nuisance. Calibration adjustments can take place at the mission beginning, but also whenever required by the user during teleoperation. This feature is deemed relevant especially on uneven terrains, where vibrations and jumps, may cause camera-robot misalignments.

Immersive visualization

Realizing a virtual 3D environment would be ineffective or not fully exploited if presented to the user through a classical 2D screen, since depth perception would be lost. This is the reason why the developed interface has been designed to be presented through immersive 3D displays, in particular VR headsets. Such devices have recently

received a considerable improvement in terms of “isolation”, image quality, field of view and comfort.

Visual aids

Visual aids are graphical elements representing, in various forms, information coming from onboard sensors. They are designed to make more intuitive not only sensor data, but vehicle’s dynamics within the remote scenario and operator’s commands as well.

Eight different visual aids are proposed, providing information either about the robot’s *surroundings* or on the *direction* to follow from driver’s viewpoint. They are:

1. *Traversable Area.* It provides a graphical representation of the expected crossable terrain area within current field of view as colored overlay on the terrain model (Figure 4.12). This kind of information is really helpful for an operator when planning the next movement, especially for outdoor hostile scenarios.

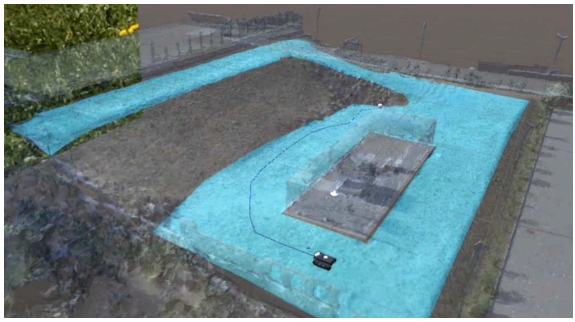


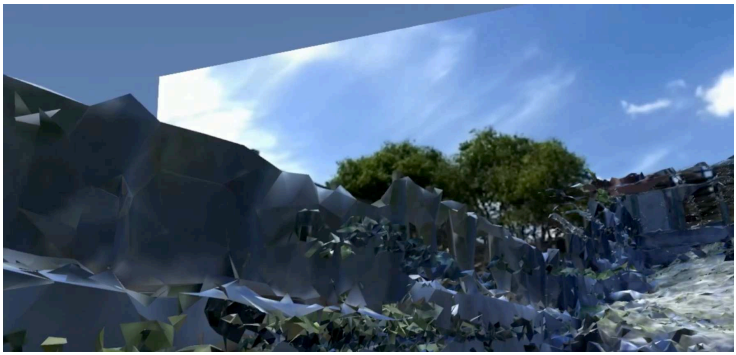
Figure 4.12 The traversable area represented as a cyan layer overlaid onto the environment model reconstruction

2. *Camera view extension.* Remote images coming from onboard cameras always suffer for the problem of providing a limited view of the environment, mainly related to their location on the vehicle and their field-of-view. In general, this implies a loss of peripheral vision. To cope with this problem, several works reported in Sec. 4.1 suggest the adoption of wide-angle cameras [45, 70, 89].

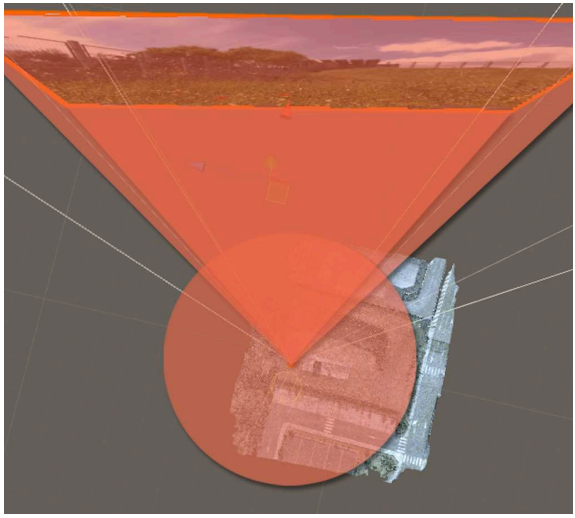
   Another common technique, named *video mapping*, consists of including live video images into an AV context. In the approach here presented computer graphics is employed to extend live streamed images with the environment reconstruction, thus allowing the operator to "see" those parts of the environment not covered by the onboard stereo camera viewport. Thanks to textures, also appearance is preserved in 3D reconstructions which makes the transition between real video images and the virtual model more comfortable and smooth for the teleoperator. Figure 4.13a shows an extended camera view example, while Figure 4.13b depicts the relative positions between video screen and 3D environment model.

3. *Top view.* It provides an overview of the vehicle's close surroundings. Furthermore it allows the operator to immediately comprehend vehicle heading and then, to make the correct decision.

4. *Exocentric View.* Depending on the specificity of the maneuver required, a third-person view, also known as exocentric view, can valuably simplify the task accomplishment, allow-



(a) Example of extended camera view



(b) Top view showing video screen position with respect to the environment reconstruction

Figure 4.13 The extended camera view visual aid

Figure 4.14 The exocentric view: two examples

ing to perform it quicker and more accurately than with an egocentric or first-person view [28, 63]. According to Nielsen *et al.* [63], adjustable perspective supports all three levels of Endsley's situational awareness (perception, comprehension, and projection) [25]. However, generally it is not immediate to recreate of such views.

In the developed interface the exocentric view is tethered to robot pose given by GPS, IMU and odometry data and reconstructed from the available 3D environment textured mesh. In this way, a large number of exocentric views can be generated. In [70] authors report exocentric 3D views slowed down task accomplishment due to the time needed by operators to define the best perspective to carry out a mission. However, the accuracy achieved with the final result motivates the usefulness of the visual aid proposed. Figure 4.14 shows some examples of reconstructed virtual views.

5. *Virtual Pointer.* This visual aid has been often proposed in the literature as a means of human-machine interaction, since

the earliest mixed reality approaches [57, 79]. The pointer is implemented as a casted ray originating from the robot and going towards the pointed area of the environment. A sort of road-sign appears at the position the ray hits the environment model, showing relative altitude, relative distance, and traversing cost as a percentage (defined with respect to the maximum cost in the map). Traversing costs are computed as described in Chapter 3. This information is displayed as plain text and provides the operator with precise values and measures. This feature allows the operator to establish a *spatial relationship* with the virtual environment, i.e. to relate to its "physical size" (similarly to the virtual introduction of common objects proposed in [61]). Figure 4.15 shows an example of virtual pointer usage during navigation.

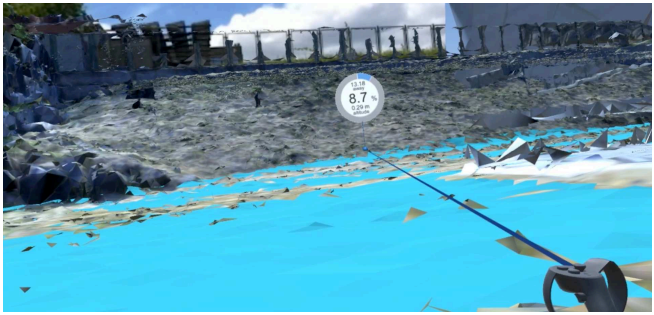


Figure 4.15 Example of road-sign like post arising from the virtual pointer including relative altitude, relative distance, and traversing cost

6. *Guide Arrow.* A 3D graphical arrow has been introduced in the interface, guiding users to follow the recommended path.

Moreover, thanks to 3D visualization in the AV context, the arrow direction is even easier to catch, thanks to the higher depth awareness provided. Arrows have proved to be particularly appreciated by unskilled operators or novices, compared to compasses or light sources, for directional indications [12, 62].

7. *Centerline*. As suggested by the standard use of lines in road markings, a centerline is adopted to suggest the ideal vehicle trajectory to reach a specific target. This path is generated through a path-planning algorithm taking into consideration traversing costs, as explained in 3.6.4. However, unlike common roads, the vehicle must stand *above* the continuous line, and not to stay within a "road lane". In fact, the centerline is intended to provide a straightforward reference to the remote driver. Figure 4.16 shows the implemented centerline.

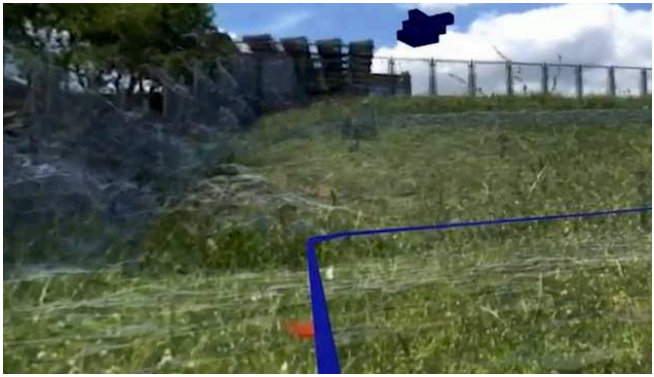


Figure 4.16 The recommended path presented as a blue centerline

8. *Robot virtual model.* As reported in [42, 50], representing the robot's chassis, both for exocentric and (partially) for egocentric views, increases situational awareness and prevents accidental collisions. The robot model replicates the vehicle's appearance, size and shape to minimize collisions.

Since multiple visual aids and the video frames can be displayed at the same time, transparency of all these elements can be adjusted, allowing the user to feel comfortable with the interface by suitably blending video and graphical elements. Guidelines by Harrison *et al.* [36] have been considered for the design of the mixed reality views.

4.3 Experimental Setup

For the experimental validation of the interface the U-Go robot has been suitably retrofitted with the sensing needed for the specificity of the task (Figure 4.17).

In particular, in addition to RTK GPS and IMU, a ZED camera by Stereolabs [106] has been adopted for stereoscopic video acquisition. This camera acquires high-resolution images (1280×720) of the environment in front of the robot at 15 fps and is connected to a Jetson TX1 NVidia board [66] which runs the ZED SDK.

The proposed teleoperation interface has been developed with the Unity3D software [94]. It contains the photogrammetric textured reconstruction of the environment and the 2D map of traversing costs is overlaid, from a top perspective, onto such reconstruction. A grid resolution of 25 cm on a size has been considered for the costmap. The driver control station of the teleoperation system con-

Figure 4.17 The U-Go robot adapted for teleoperation missions

sists of a PC, remotely connected to the U-Go ROS network. In fact, the ROS framework allows to set up remote connections, based on TCP-IP protocol. The operator wears an Oculus Rift headset [67] which provides an immersive view of the virtually generated environment, while driving commands to the vehicle are issued through an Oculus Touch controller. It is used for the calibration process as well. Data exchange between the Unity3D software and the ROS network is carried out by Unity3D scripts sending driving commands and retrieving sensor data (to and from the robot platform) through ROS nodes specifically developed for this task.

During the mission, acquired video frames and pose data of the robot are collected and chronologically related through timestamps included in the sensor message format provided by ROS. The most recent couple of sensor data and streamed images are used to update

operator's view. In particular, video frames are mapped on virtual planes located in front of the robot. Figure 4.18 shows a scheme of the described data flow.

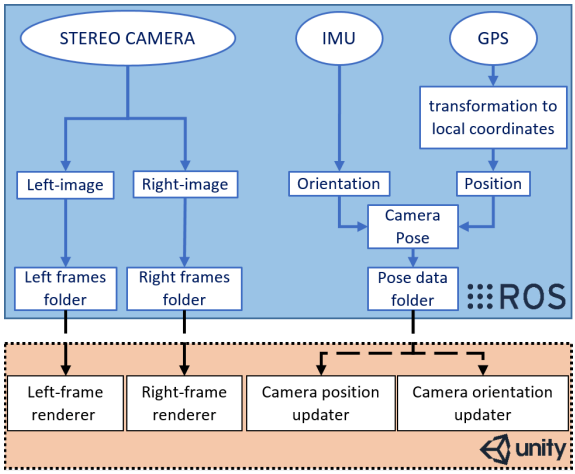


Figure 4.18 Data flow involved in the interface implementation

4.4 Interface testing

Once the whole system was setup, its operation and effectiveness have been assessed through a *pilot user study*, in addition to the over thirty runs during system development and tuning. After few minutes of briefing and warm-up session, seven users were asked to teleguide the robot across the testing area described in Sec. 3.6.4 (i.e. the landslide-like area shown in Figure 3.9), for a couple of times. In this case, in addition to the already known challenges of the terrain (different slopes, rough ground surface, presence of non-traversable

regions), the rich vegetation often caused a limitation in visibility through the camera.

Through the pilot study, the interface usability to display multiple information sources (via the proposed visual aids) has been tested, along with the management of operators' control commands. During all the testing sessions, all data exchange is assumed to rely on a stable Internet connection, free of latency issues. In particular, the management of sensor and model information has been tested (GPS, IMU, camera and graphical 3D model), because misalignments between reality and virtuality were expected to be a main issue. In the current developed implementation, two types of misalignment may arise, namely *pose misalignment* (actual vehicle attitude versus the one represented on the operator's interface), and *sensor misalignment* (discrepancies among different sensor information displayed in the visual interface).

During tests, although the information flow was not perfectly synchronized (since sensor data are essentially used with a "first in, first served" policy), occasional inconsistencies between sensor data and the environment model have not been judged critical by testing users, whilst most of the time all data have been received and displayed coherently. This is likely to be related to the relatively low maximum speed of the vehicle ($\sim 0.8\,\mathrm{m/s}$).

In particular, sensor misalignments are chiefly caused by the different acquisition speeds. In fact, while the graphical model is pre-loaded on operator's control station (and therefore no latencies in views generation can occur), delays were all related to incoming sensor information. In particular, the GPS rate is 500 ms, the

IMU is 100 ms, while the camera frame rate is set at 15 fps. Furthermore, packet loss occasionally occurred through the network, thus determining sudden motions of both the view and the visual aids. Probably GPS low update rate is the weak link; however this issue is planned to be overcome by employing a Kalman Filter, to combine acquired positions with odometry, thus mitigating the criticality of GPS data intermittency. In any case, the performed testing has confirmed the proper working of both the robot hardware and the ROS network.

4.4.1 Visual Aids evaluation

Both "regional" and "directional" visual aids have been generally deemed helpful in displaying sensor data in an intuitive manner. In particular, although in its quite basic implementation, the *centerline* has been the most needed aid; besides providing good ground visibility, it clearly indicated the path to follow as a sort of sequence of landmarks. Another visual aid appreciated by drivers has been the *traversable area*, along with the *extended camera view*. The latter, according to users' comments, effectively enhanced sense of immersion thanks to the involvement of operator's peripheral vision.

Most users have needed some adaptation time when switching from the egocentric to the *exocentric view*, which, however, has been still commented as very helpful. Also the *robot chassis*, which can be clearly seen while in third-person view, has been appreciated although some improvements, in terms of realism, have still to be performed (e.g. tracks movement). While stopped, most of the users has easily tried out the *virtual pointer*, whereas, whenever used dur-

ing navigation, it has occasionally caused view occlusions, if the road-sign was in very close areas in front of the vehicle. This issue calls for further corrections. The particular 3D representation of the *directional arrow* turned out to be really intuitive during navigation. However, some suggestions on adapting arrow heading according to vehicle speed have been raised. In fact, the current implementation is based on a dynamic and sequential selection of the checkpoints composing the centerline. Moreover, arrow color and position within the screen has been suggested to be customizable by the user, in order to adapt it with respect to the color background or, in general, to fit with user's expectation and comfort. Surprisingly, the *top view* turned out to be the least appreciated visual aid, since many operators have referred no use of it. Despite this, some operators considered the top view more immediate to get an overview of robot surroundings with respect to switching to an exocentric view.

4.4.2 Overall evaluation of the interface

Operators have unanimously judged positively the full immersion and isolation provided by the control station. Image vibration due to the ground irregularity and lack of image stabilization has not represented a critical issue. Color hues and the specific colors chosen have properly blended with the considered scenario, despite the large amount of vegetation. Figure 4.19 shows a view caught during one of the trials, including a large number of the visual aids proposed.

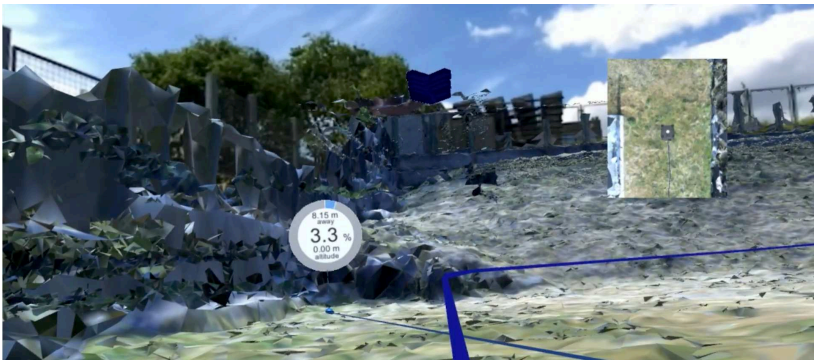


Figure 4.19 A view through driver's HMD, which includes: blue centerline, informative post (associated to virtual pointer), environment top view (top-right corner), directional indicator (top-center), extended view, MR context (live video frames in the background)

4.5 Conclusions

A novel MR-based interface for robot teleoperation has been described, conceived to improve remote driving performances by enhancing user's situational awareness and intuitively communicating sensor data along with both streamed video from the remote area and 3D environment model. A three-dimensional virtual environment has been considered to combine all the elements above, thus obtaining an interface with the following main features: single-frame integrated view, immersive visualization, and visual aids. Eight different VAs have been described: traversable area, extended camera, top view, exocentric view, virtual pointer, guide arrow, centerline, and robot's chassis. The proposed approach has been implemented and tested on a real system made up of: a UGV, with a stereo camera

and localization sensing, and the driver's control station, including a desktop computer, presenting the interface via VR headset and a controller for issuing commands.

System tests have been performed since initial trials and then a pilot user study on an outdoor unstructured terrain has been carried out. Results are very encouraging because of the positive judgments by all of the users. Most of the proposed visual aids have proved to be very effective and helpful. Some of them have been highly appreciated, others still call for some improvements, according to users' suggestions. The outcomes on uneven terrain trials have demonstrated the advantage of the intuitive and integrated visual communication proposed, of the effective graphical alignment and of the exo-centric perspectives. Moreover, results proved that discrepancies among sensor acquisition timings do not worsen situational awareness, as long as a sufficiently-large communication bandwidth is guaranteed. Clearly, a wider set of tests have still to be carried out, in order to derive statistical metrics on the interface usability.

To sum up, the potential of MR in robotics teleoperation in the outdoors has been investigated. Many different alternatives and enhancements can be explored thanks to the mainly unlimited variety of possibilities achievable in the virtual context. Moreover, the recently strengthened VR technology valuably supports this approach thanks to immersive visualization.

Chapter 5

Complete coverage path planning for aerial survey

In rough or risky scenarios, the problem of having a map or at least an approximate representation of the environment to get through with a ground vehicle is extremely challenging, before the navigation is actually performed. To this end, the adoption of *a flock* of UAVs to survey and reconstruct 3D models of the environment can be really helpful, especially for traversability analysis purposes. In particular, photogrammetric 3D reconstruction from airborne imagery has gained more and more relevance over the years, thanks to the enhanced quality of the results and the increased computing power available [21, 93].

In this chapter a complete solution for the problem of aerial photographic scan for 3D reconstruction is presented, addressing two main issues. The first is the so-called *Complete Coverage Path Plan-*

ning (*CCPP*) and concerns the problem of defining UAV trajectories for photogrammetric aerial image acquisition. In CCPP, a path that fully covers a *Region of Interest* (*ROI*) is defined, while still avoiding possible obstacles. When no-fly zones are present, a suitable decomposition into subregions of the whole area is performed and then, for each subregion, *back-and-forth paths* are computed. The second issue is related to the employment and the management of a *flock of UAVs*, to speed up the coverage mission through a parallel execution. Two different solutions to assign the subregions among the flying vehicles will be presented: one minimizing the path length of the whole flock in reaching the different regions, the other trying to sub-optimally balancing the whole coverage path length among all flock members.

The overall approach has been implemented as a software framework, whereby it is possible to create and test different situations for the flock management and, eventually, to obtain *waypoint lists* from the computed trajectories, in order to easily test the results with a team of multi-copters.

5.1 Complete coverage path planning taxonomy

A large number of applications require the surveying of wide areas such as industrial automation, environmental monitoring, and disaster response (earthquakes, landslides, avalanches). Robotics proved to be very helpful in these tasks, often risky as well as tedious for human operators. In this case, CCPP algorithms have to be employed.

According to the application, the ROI to cover can be an *open surface*, such as, for instance, in photogrammetry [21, 93], and in urban structure coverage [18]. In other applications it can be represented as a *closed surface* such as in spray-painting [5], and in underwater inspections [26].

Furthermore, according to [43], CCPP algorithms can be classified into:

- *offline algorithms* if the geometry of the environment is known and constant during the mission, and paths are pre-planned
- *online algorithms* if the model of the environment is built and updated by live acquisitions of the robot, while sweeping the region to cover

In offline algorithms, *decomposition techniques* are used to avoid possible obstacles, or non-coverable zones, within the ROI, namely approximate, semi-approximate and exact cellular decompositions. Therefore, the three main features of an *offline CCPP algorithm* are [43]:

- *the decomposition technique*
- *the backtracking mechanism* to determine the linking paths among the subregions
- *the optimality factors* considered in trajectory generation

Although coverage path planning has been equally investigated for ground, aerial or underwater vehicles, in the following sections trajectory computation for UAVs is discussed.

5.2 Complete coverage path planning for aerial vehicles

Unmanned aerial vehicles, especially multi-copters, are particularly suitable for survey applications. Their use is further motivated by the recent improvement in performances and the continuous decrease in costs, size and weight. For all of these reasons the deployment of UAV flocks has also been proposed in the related literature, as they can carry out coverage missions more efficiently and quickly than a single vehicle, while increasing system robustness and flexibility in the solution.

In [21], the authors developed a strategy for coverage path planning of a *single vehicle* taking into account the UAV's power consumption, by means of a model identified from real measurements. In particular, they found out that in the back-and-forth motion, turns are the main cause of loss of time and energy. Similar conclusions have been reported also in [30, 47, 93]. This is because a turn involves many steps: deceleration, stop, reaching of the subsequent row, and, again, acceleration. Therefore, it is crucial to choose the trajectory with the minimum number of turns, whenever possible. Another CCPP solution designed for a single UAV is described in [93], where the *optimal line sweep direction* is defined as the one ensuring coverage paths with the lowest number of turns. Furthermore, after the decomposition step, coverage paths' directions within each subregion are chosen in such a way to minimize the distance to reach the subregion itself.

Considering now the solutions for complete coverage path planning of *flocks of UAVs*, an early work is presented in [97], proposing a *formation-based* strategy. The agents, arranged in line, sweep the region of interest, which is assumed to be free of obstacles or non-coverable zones. A relevant multi-UAV solution for aerial imaging in precision farming is described in [7]. This solution includes: task negotiation, georeferencing management, optimal path planning, and maneuverability enhancement of the quadrotors thanks to a customized flight control. Approximate cellular decomposition is proposed for no-fly zone management. A pretty unique work in the literature which takes into account environment morphology is [74]. In particular, it consists in a multi-agent methodology for 3D coverage paths obtained by superimposing the computed paths to the digital elevation model of the area to survey. An approximate hexagon-based decomposition is proposed. The obtained hexagonal cells are then grouped and assigned to each UAV.

It can be noticed that the employment of several UAVs for CCPP is still not a fully explored area in current robotics research. Some works lack of no-fly zone management [97] or consider only planar coverage paths, ignoring environment geometry (i.e. at a fixed altitude above the ground) [7, 97]. Moreover, approximate cellular decomposition have often been adopted [7, 74].

The approach proposed in this chapter covers the gaps of the individual works briefly reported above. In particular, the main contribution of such approach is the development of a software framework for the flock management and testing for the CCPP problem.

The framework integrates, *in a single solution*, the main features separately proposed in the related literature, namely:

- no-fly zones handling via *exact* decomposition of the ROI
- optimal path choice within each subregion according to power consumption and geometry consideration (if available)
- negotiation for subregion/UAV assignments
- *Geographic Information Systems* (*GIS*) management
- communication with real UAVs' autopilots

In this way, an end-to-end GIS software tool has been realized, allowing to handle any aspect of the flock survey mission: from no-fly zones and georeferenced maps, to the deployment on real hardware. The details of the integrated features will be given in the following sections, along with the motivations behind the implementation choices.

5.3 Morse decomposition

The present work falls into the category of *offline* complete coverage path planning algorithms. Therefore, the first step is to define the ROI to survey as a 2D region on the environment map. If a coarse DEM of such environment is considered, the ROI is defined as a 2D top view over such model. After that, *no-fly zones* have to be identified, i.e. those zones where UAVs are not allowed to fly above. Both the ROI and the no-fly zones are defined by sequences of vertexes, thus

obtaining *polygonal zones*. At this point, the whole area is decomposed into free-to-fly subregions via a *Morse-based decomposition algorithm*, as reported in the following.

The Morse decomposition [2] is a method for *exact cellular decomposition* based on the analysis of Morse functions' singular points. The concepts of *differentiable function*, *Hessian matrix*, *singular point*, and *Morse function* are here recalled, in order to provide an overview of the theoretical background of such decomposition method. Given an arbitrary real-valued multi-variable function $h : R^m \rightarrow R$, its *differentiable function at* $p \in R^m$ is

$$dh(p) = \left[\frac{\partial h}{\partial x_1}(p), \ldots, \frac{\partial h}{\partial x_m}(p) \right] \tag{5.1}$$

The *Hessian matrix* $H(h)$ of a real-valued multi-variable function $h(x_1, x_2, \ldots, x_m)$ is a matrix whose elements are all the second derivatives of h, sorted with respect to the variables

$$H(h) = \begin{bmatrix} \dfrac{\partial^2 h}{\partial x_1^2} & \dfrac{\partial^2 h}{\partial x_1 \partial x_2} & \cdots & \cdots & \dfrac{\partial^2 h}{\partial x_1 \partial x_m} \\ \dfrac{\partial^2 h}{\partial x_2 \partial x_1} & \dfrac{\partial^2 h}{\partial x_2^2} & \cdots & \cdots & \dfrac{\partial^2 h}{\partial x_2 \partial x_m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 h}{\partial x_m \partial x_1} & \dfrac{\partial^2 h}{\partial x_m \partial x_2} & \cdots & \cdots & \dfrac{\partial^2 h}{\partial x_m^2} \end{bmatrix} \tag{5.2}$$

A value $p \in R^m$ such that all its derivatives are null is a *singular point*, namely

$$\frac{\partial h}{\partial x_1}(p) = \ldots = \frac{\partial h}{\partial x_m}(p) = 0 \tag{5.3}$$

Furthermore, the p singular point is *non-degenerate* (i.e. *isolated*) if and only if its Hessian $\left(\frac{\partial^2 h}{\partial x_i \partial x_j}(p)\right)$ is non-singular. Finally, a multi-variable function whose singular points are all non-degenerate is referred to as a *Morse function*.

Cell decomposition is obtained by sweeping a *slice* through the whole target space [2]. In particular, the slice is a co-dimension one manifold defined in terms of the pre-image of a real-valued Morse function, $h : W \rightarrow R$, where W is the space (2D in this case) to be covered. Whenever the slice reaches the Morse function's singular points its *connectivity* changes. According to the Morse function chosen, the slice can have different "shapes", thus obtaining different decomposition patterns.

In this work, the so-called *Morse-based boustrophedon decomposition* has been adopted, which is a planar decomposition and, in particular, both *horizontal* and *vertical planar decompositions* have been considered. This choice allows to obtain a fair final decomposition with a straightforward implementation. The vertical linear slice is obtained by choosing as Morse function

$$h(x, y) = x \tag{5.4}$$

while the horizontal linear slice is related to the following Morse function

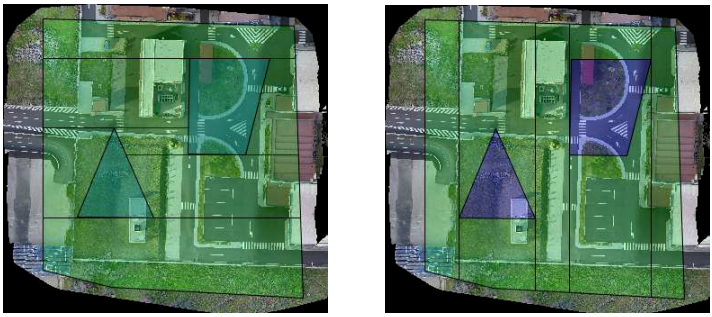
$$h(x, y) = y \tag{5.5}$$

The former sweeps the target space from left to right, i.e. along the x-axis, while the latter from bottom to top, i.e. along the y-axis, in the xy plane. Formally, the slice, either vertical or horizontal, is given

by the pre-image of this Morse function, i.e. $W_\lambda = h^{-1}(\lambda)$. The slice is parameterized by $\lambda \in R$ which determines its location in the target space. By increasing the value of λ, the slice sweeps from left to right through the workspace (in the vertical case). Similar considerations can be done for the horizontal case. As the slice sweeps the space, it could intersect obstacles, and the slice itself is divided into smaller segments (i.e. slice connectivity increases). Furthermore, as the slice leaves an obstacle region, the segments merge again into larger segments (i.e. slice connectivity decreases). The points where these connectivity changes occur are singular points for the Morse function considered. In the following, since singular points arise at specific points belonging to obstacles boundaries, they are also referred to as *singular points of the obstacles*, by slightly abusing notation.

It is worth remarking that, since georeferenced maps are considered as 2D target space in this case, x coordinates are easting coordinates, while y coordinates are northing coordinates, in accordance with the UTM convention (see Sec. 2.2.1). Figure 5.1 depicts the final results of the linear decomposition along the vertical and the horizontal axis performed on a map.

In order to implement the planar decompositions described, some geometrical considerations have been exploited, arising from the specificity of the Morse functions adopted. First, singular points of the no-fly zones are detected (Figure 5.2b). After that, for each singular point a line is considered, namely the line passing through the considered singular point and parallel to the slice. These lines cover all the width of the area to be decomposed and if one of these lines intersects a border of any no-fly zone (including the one which



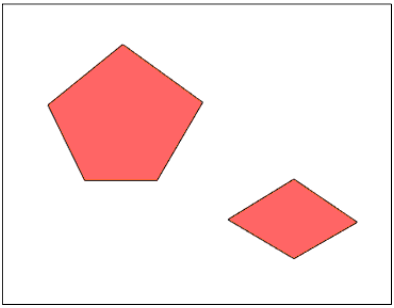
(a) Horizontal decomposition (b) Vertical decomposition

Figure 5.1 Graphical examples of Morse-based decompositions

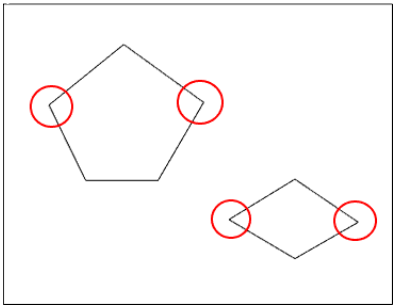
the singular point belongs to), it breaks at the intersection closer to the singular point. Once those lines are defined, they are split at the singular points into shorter segments. The perimeter of the whole 2D area and the perimeter of inner no-fly zones are divided into segments as well, either at singular points or corners (Figure 5.2c). Eventually, adjacent segments (i.e. those sharing a common vertex) are linked, thus obtaining the subregions to be covered (which are essentially closed polygonal shape). Figure 5.2 reports all the steps performed in the vertical case described above in detail.

5.4 Trajectory generation

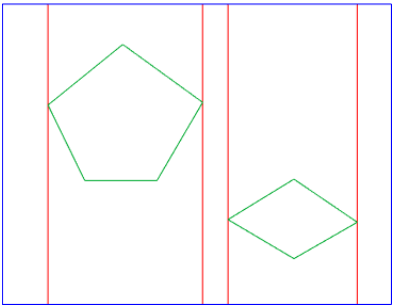
Once subregions have been identified, coverage trajectories have to be defined. In the general problem of CCPP, trajectories should imply basic motions, for simplicity in control [17]. The two most frequent patterns in the literature are the *back-and-forth* (or *lawnmower* or



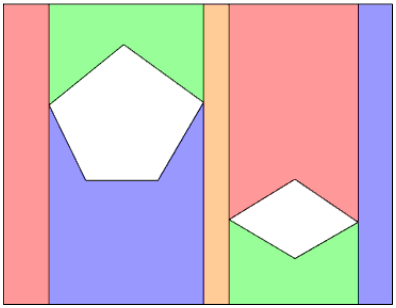
(a) Starting situation (no-fly zones in red)



(b) Singular points



(c) Lines and perimeters splitted into segments

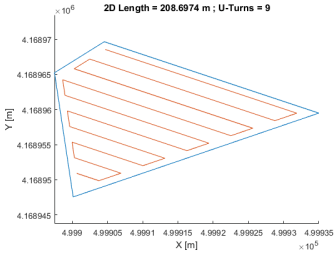


(d) Decomposed coverage area

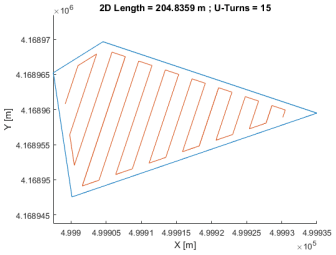
Figure 5.2 Morse decomposition steps

boustrophedon) motion and the *spiral* motion. These trajectories have been usually preferred because of their flexibility of adaptation to even rather complex shapes. In this work, as coverage pattern the *back-and-forth motion* has been considered, since the resulting paths end close to the boundaries of the subregion, rather than in the center (as in the spiral case). This is preferable whenever the same UAV has to cover several regions as explained in Sec. 5.5. For each subregion, $2n$ planar trajectories are derived, where n is the number of edges of the considered subregion. In particular, both parallel and perpendicular trajectories with respect to each edge are generated, as shown for a trapezoidal area in Figure 5.3.

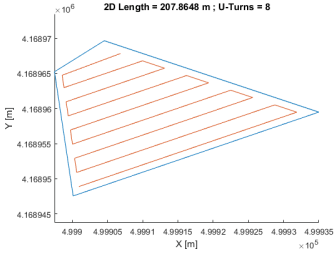
At this point, the optimal trajectory is chosen as the one with minimum number of turns, thus minimizing time and energy spent during the back-and-forth path execution (see Sec. 5.2). Furthermore, if the DEM of the environment is known, 3D trajectories are derived from the 2D ones already described. In fact, the altitude value is assigned considering the ground height values (retrieved from the elevation model) and adding a safety margin altitude. Moreover, in order to derive a smooth path, the DEM is previously dilated, thus further increasing the distance from the ground (Figure 5.4). In this case, the optimal trajectory is the one with the shortest *3D length*, among those paths with the minimum number of turns. Therefore, at the end of this step, the coverage paths for each subregion are defined. An example of optimal subregion trajectories definition for a certain ROI is shown in Figure 5.5. As it can be seen, the orientation of coverage paths can change from subregion to subregion, in order to fulfill the optimality criterion.



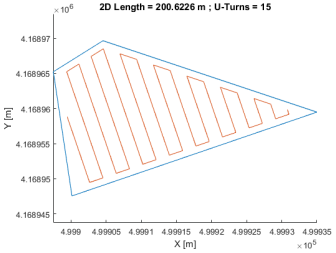
(a) Parallel to 1st edge



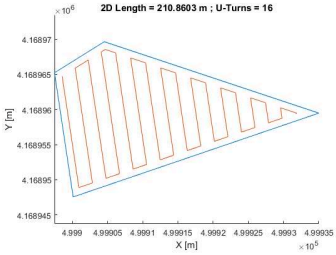
(b) Perpendicular to 1st edge



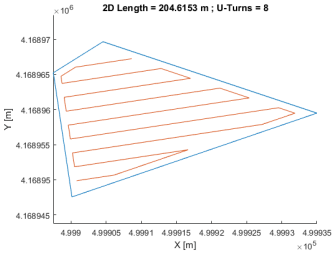
(c) Parallel to 2nd edge



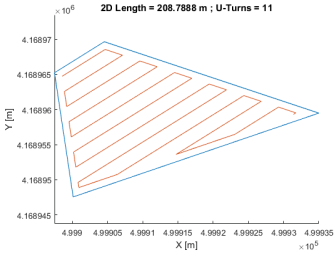
(d) Perpendicular to 2nd edge



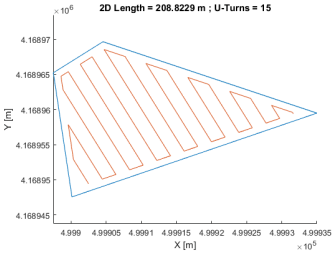
(e) Parallel to 3rd edge



(f) Perpendicular to 3rd edge



(g) Parallel to 4th edge



(h) Perpendicular to 4th edge

Figure 5.3 Example of generated trajectories for a trapezoidal area (parallel and perpendicular to edges)

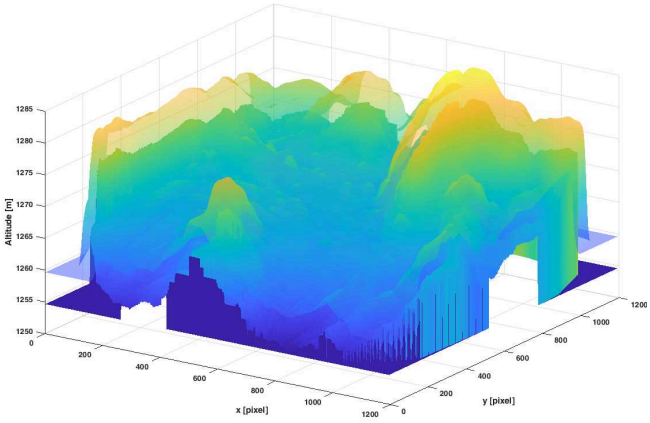


Figure 5.4 Original DEM and the dilated version in transparency

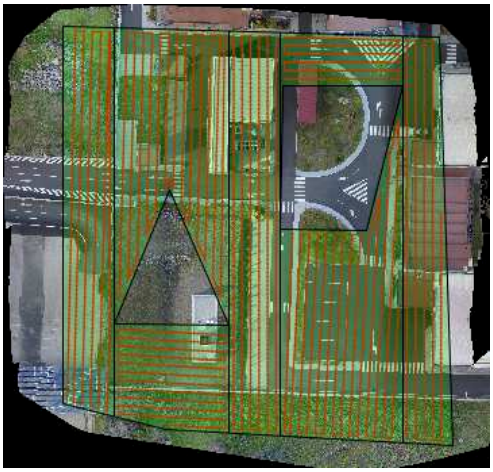


Figure 5.5 Example of environment decomposition with optimal coverage trajectories

5.5 Coverage subregions assignment

The proposed approach exploits the use of a flock of UAVs to parallelize the survey mission. This implies the definition of a rule to assign subregions (and their related paths) to each UAV, once UAVs starting positions are specified on the map. Two different policies have been developed.

The *first policy*, described in [34], assumes a number of vehicles equal to the number of subregions to be covered and consists in defining the paths from each UAV to each subregion, with a 3D implementation of the A* algorithm, and evaluating the lengths of the computed paths. In doing that, both the two ends of each subregion's coverage trajectory are considered as potential target positions. In fact, as underlined by [93], coverage trajectories can be traveled indistinctly along either one of the two running directions. Therefore, for each subregion, the path lengths to reach the two ends of the back-and-forth trajectory are compared and the shortest one is chosen, thus establishing the running direction. Algorithm 2 reports the steps to perform this choice for the entire flock. After that, all the possible UAV/subregion combinations are derived and the combination with minimum *global* path length for the whole flock is chosen. Such global path length, for a certain combination, is simply given by the sum of the trajectory lengths of each UAV.

The first policy identifies the optimal combination, minimizing the distance traveled by the whole flock, only for the first part of the coverage mission, i.e. the *subregions reaching*. The *second policy* is realized via a *competition routine* which aims at equally distributing the lengths traveled by each UAV instead, by taking into account

Algorithm 2 first_last_waypoint function

for j = 1 to length($\text{traj}_{\text{left}}$) **do**
  for i = 1 to n_{drones} **do**
    path_1 = A*_plan(UAV_i position, first waypoint traj_j)
    path_2 = A*_plan(UAV_i position, last waypoint traj_j)
    if 3Dlength(path_1) < 3Dlength(path_2) **then**
      start_to_first(j, i) = path_1 ▷ shortest path for UAV_i to traj_j
    else
      start_to_first(j, i) = path_2
    end if
  end for
  length_path(j,i) = 3Dlength(start_to_first(j, i))
end for

coverage trajectory lengths as well. In this case the *total path length* for each UAV of the flock is computed. At the beginning, this value is initialized as the distance to each subregion (computed as explained above) plus the related coverage pattern length. This sum is indicated as *trajectory length*.

At this point, the actual *competition routine* is run a number of times equal to the number of UAVs:

1. find the UAV with minimum *total path length*
2. assign the subregion to the related UAV
3. remove both the subregion and the UAV from the competition

If the number of UAVs is smaller than the number of subregions, there will be regions not yet assigned. In this case the previous routine is re-iterated, until all the coverage regions are assigned. However, at the beginning of each iteration, each UAV position is updated at the final

position of the previously assigned coverage path and then the A* algorithm is reapplied from these points to the coverage trajectories left. The new *total path length* value for each UAV is updated as *total path length* = *traveled distance* + *trajectory length*, where *traveled distance* takes into account the lengths of assigned paths until then. This procedure ends when all the coverage regions are assigned, as reported in Algorithm 3.

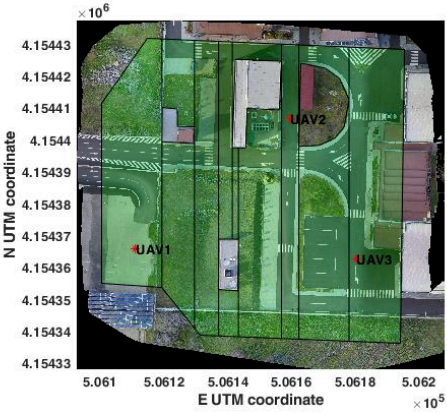
Algorithm 3 Flock competition algorithm

```
traj_assigned = []
traj_left = [1:traj_num]
traj_total = [1:traj_num]
[start_to_first, total_path_length] = first_last_waypoint(traj_left, start)
update total_path_length with the trajectories length
while traj_assigned != traj_total do
    for i = 1 to n_drones do
        if traj_assigned == traj_total then
            break
        end if
        (j, i) = find the minimal entry of total_path_length
        assign trajectory_j to UAV_i
        remove trajectory_j and UAV_i from the competition
        update traj_assigned
    end for
    build or update the final_trajectories
    update UAVs current positions
    update traj_left
    if traj_assigned != traj_total then
        [start_to_first, total_path_length] =
                                first_last_waypoint(traj_left, current_positions)
        update total_path_length as traveled_distance + trajectories_length
    end if
end while
```

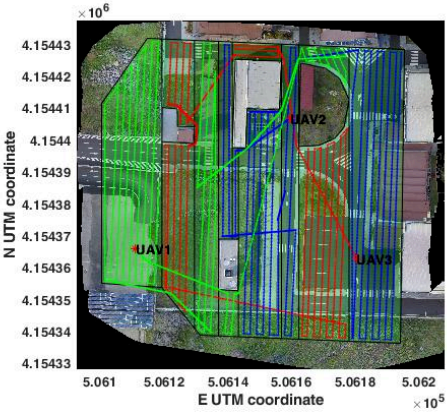
Clearly, the competition routine does not guarantee that the distance globally traveled by the whole flock is the minimum. In this sense this second policy is considered *suboptimal*, although it has been already adopted in the literature [60]. However, it is an acceptable tradeoff between achieved results and computational burden. In fact, the problem of computing all the possible combinations among UAVs and coverage trajectories becomes nearly intractable as the number of subregions or vehicles increases, especially if there are fewer UAVs than regions to be covered. In this case further combinations have to be considered, i.e. the ones related to the successive iterations. Finally, the second policy is inherently suboptimal in terms of path lengths distribution, because the final result depends on the lengths of coverage trajectories, whose values can widely vary.

A further feature of the subregion assignment solution is the capability of handling also the situation in which the UAVs are more than the subregions to be covered. In this case, the longest coverage path (among the ones available at each iteration) is iteratively halved, until the number of UAVs equals the number of subregions. At this point, both the policies described above can be applied normally.

Finally, computed trajectories are formatted as a *waypoint list*, ready to be delivered to the Ardupilot firmware [4] for UAV autopilots. This allows to easily test the results with a real flock of aerial vehicles. A coverage mission of 12 subregions with a team of 3 UAVs is shown in Figure 5.6. Moreover, since the DEM of the area has been included, 3D trajectories have been computed, which are shown separately in Figure 5.7, along with the linking paths between the different subregions.

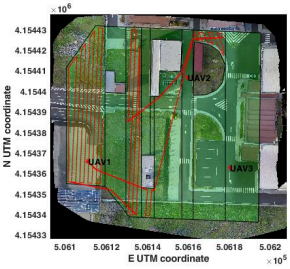


(a) Flock starting positions (subregions in green)

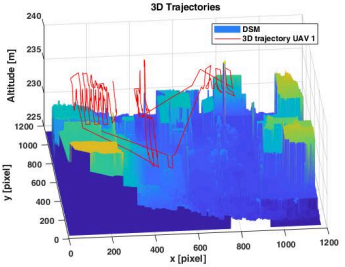


(b) 2D top view of flock coverage trajectories
(UAV1 in green, UAV2 in blue, and UAV3 in red)

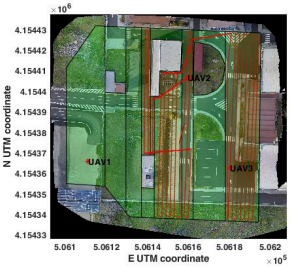
Figure 5.6 Example of planned paths with a flock of 3 UAVs and 12 subregions to be covered



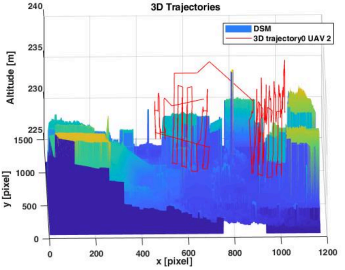
(a) 2D top view of UAV1 trajectory



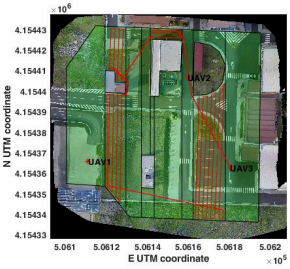
(b) 3D trajectory of UAV1



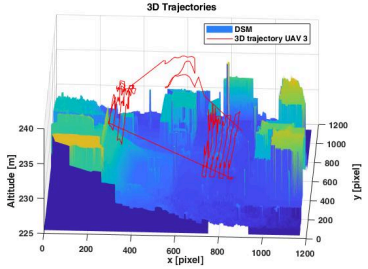
(c) 2D top view of UAV2 trajectory



(d) 3D trajectory of UAV2



(e) 2D top view of UAV3 trajectory



(f) 3D trajectory of UAV3

Figure 5.7 Example of final coverage trajectories for a flock of 3 UAVs

5.6 Results

In this section the results achieved in testing the proposed flock management system, described in Sec. 5.5 are reported. In particular, tests have been performed with the *second policy*, paying particular attention on the balance in path lengths distribution among the UAVs. The second policy has been preferred to the first one because the former, despite being suboptimal, manages the overall coverage mission and not subregions reaching only (as the first policy). Therefore, the second policy was expected to provide more interesting results.

Furthermore, in order to consider the most general case, the environment geometry is assumed to be known, namely the digital elevation model. The external area of the engineering department's laboratories has been considered as area to survey (see Sec. 3.1). The no-fly zones (in blue) and the result of the decomposition (coverage subregions in green) are shown in Figure 5.8.

In order to test a variety of conditions wide enough to perform a statistical analysis, 60 trials have been performed with 6 flocks (with a number of UAVs ranging from 3 to 8) and a set of 10 different *starting distributions* (e.g. gridded, linear, chessboard-like, etc.). In particular, the same starting formations have been considered for each of the 6 flocks, in order to obtain results which could be comparable as the number of UAVs in the flock changes. The established starting formations are shown in Figure 5.9 and 5.10: the 5-UAV flock is depicted.

As the number of UAVs increases, the *average length*, i.e. the length traveled by a vehicle of the flock *on average* is expected to

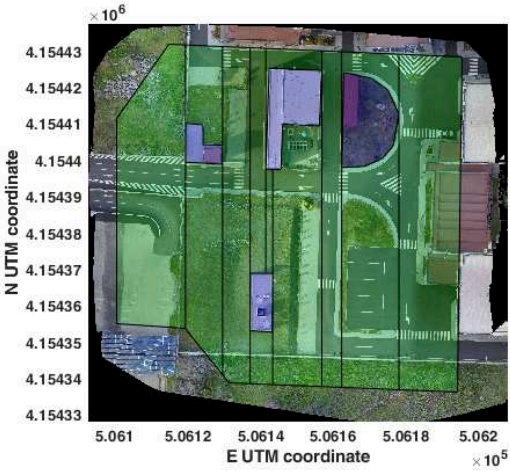


Figure 5.8 Coverage subregions (in green) and no-fly zones (in blue) of the testing area

decrease, while the *difference between longer and shorter lengths* within the same flock is expected to increase. The latter implies an increase in the imbalance over the flock, which is mainly related to the number of vehicles. In fact, this number is not, in general, an integer multiplier of the number of subregions to be covered and, therefore, some UAVs will cover more regions than others (if the former are fewer than the latter). This results into an increase of the path lengths *individually* covered by these vehicles that, in addition to the back-and-forth 3D motion, will include also the linking paths to move between the subregions. However, if the UAVs are many more than the subregions to be covered, the split of longer paths could lead, in principle, to a fair distribution of lengths to be traveled by UAVs. On the other hand, the price of this solution lies in all the

issues related to the large number of vehicles involved (higher costs, difficult control, etc.).

Results obtained for the 6 different flocks, over the 10 experiments, are reported in Tables 5.1 to 5.6. Both minimum and maximum lengths assigned for each experiment are presented, along with the difference between such lengths (in order to evaluate the balance in coverage length distribution). Moreover, the mean lengths traveled by a generic vehicle of the flock, for each of the 10 experiments, are listed.

Inspection of the tables shows that (for the considered environment with its specific no-fly zones) by adding a single UAV to the flock a decrease in the mean lengths is obtained, as expected ($\sim 100\ m$ less per UAV). However, it is not possible to identify a starting formation ensuring a minimum average length. This is evident by looking at the results achieved with the different flocks for the same experiment (i.e. with a different number of UAVs at same starting formation). In fact, the average length turned out to be changing as initial positions changed. On the other hand, the best case, in terms of balanced distribution of path lenghts among all the UAVs, would be the one with the minimum deviation between the maximum and the minimum lengths for each experiment. However, similarly to what has been said above for the mean lengths, there is no starting formation which may be preferred to others, as the obtained results widely vary from experiment to experiment.

In the experiment 3 only (wide linear arrangement), deploying 6 UAVs, the best behavior has been achieved, both in terms of minimum global path length for the flock and the best path length balance.

(a) Starting positions for experiment 1

(b) Starting positions for experiment 2

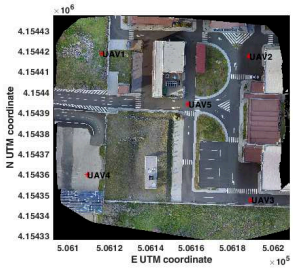
(c) Starting positions for experiment 3

(d) Starting positions for experiment 4

(e) Starting positions for experiment 5

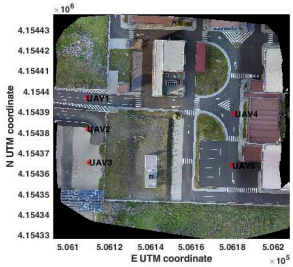
(f) Starting positions for experiment 6

Figure 5.9 Starting positions for experiments from 1 to 6 for a flock of 5 UAVs



(a) Starting positions for experiment 7

(b) Starting positions for experiment 8



(c) Starting positions for experiment 9

(d) Starting positions for experiment 10

Figure 5.10 Starting positions for experiments from 7 to 10 for a flock of 5 UAVs

This result is likely to be related to the even initial distribution of the vehicles, spanning the entire width of the area of interest. Under this condition, the competition algorithm described in the previous section was capable to define an optimal result, compared to other experiments.

Furthermore, an evaluation of the algorithm outcomes with respect to the change of starting positions has been performed by computing the standard deviation of the array of the average lengths for each flock, over the different experiments (Table 5.7). From this perspective, the best result is obtained with a team of 7 vehicles, as in this case the inner variability of the average length turned out to be only 3.8 m. In a similar way, the standard deviations from the array of the differences between maximum and minimum lengths, over the 10 experiments, have been evaluated, and reported again in Table 5.7), in order to quantitatively assess the best balance as the number of UAVs changes, regardless of the vehicles' starting positions. From this other perspective, the best result is achieved with 5 UAVs, with the minimum standard deviation (13.41 m) obtained. This standard deviation analysis highlights the relevance of the UAVs/subregions ratio. In all the considered experiments, the number of regions to survey is kept constant at 12. The most *balanced* result has been observed with 5 UAVs, whereas the most *efficient* outcome has been obtained with 7 UAVs. However, focusing on single experiments, the best trade-off between the two behaviors has been observed in an experiment with 6 UAVs, i.e. the mean value of number of vehicles for which the two (individually suboptimal) behaviors have been found. Therefore, increasing the number of vehicles would likely bring to

a further worsening in the balance distribution, thus discouraging the need to deploy a flock with more vehicles (although the average length would further decrease).

It is worth remarking that in the presented results, an acceptable behavior has been found even though the number of vehicles was not equal to the number of subregions. By increasing the number of UAVs the overall system performance is expected to worsen instead. This observation is a further motivation of the goodness of the results achieved. In fact, common intuition would lead one to think that the best and easiest solution would be to deploy always a flock with exactly the same number of UAVs as the regions to survey. However, this is misleading because most of the results depend on the specificity of the no-fly zones geometry and, then, of the decomposition considered. In particular, the best result was achieved with 6 UAVs, which is exactly half the number of the regions to cover.

5.7 Conclusions

In this chapter the problem of complete coverage path planning has been presented and a solution employing a flock of unmanned aerial vehicles has been discussed. This kind of path planning is extremely useful in 3D photogrammetric reconstruction. Such reconstruction is, in turn, crucial for geometry-based terrain traversability analysis for UGV autonomous navigation. The solution has been implemented as an integrated software framework, providing the user with a number of tools allowing to:

Exp.	Min length (m)	Max length (m)	Max-Min (m)	Mean length (m)
1	1.0036e+03	1.7075e+03	703.9	1.3243e+03
2	1.0243e+03	1.6796e+03	655.3	1.3223e+03
3	1.0455e+03	1.6829e+03	637.4	1.3298e+03
4	1.0100e+03	1.6932e+03	683.2	1.3197e+03
5	1.0534e+03	1.6823e+03	**628.9**	1.3323e+03
6	1.0081e+03	1.7013e+03	693.2	1.3266e+03
7	1.0630e+03	1.6934e+03	630.4	1.3447e+03
8	1.0331e+03	1.6809e+03	647.8	**1.3189e+03**
9	1.0384e+03	1.7065e+03	668.1	1.3360e+03
10	1.0677e+03	1.7092e+03	641.5	1.3526e+03

Table 5.1 Experimental results with a 3-UAV flock

Exp.	Min length (m)	Max length (m)	Max-Min (m)	Mean length (m)
1	667.0071	1.4426e+03	775.5929	1.0124e+03
2	718.7880	1.4172e+03	698.412	997.6131
3	721.4137	1.4189e+03	697.4863	**995.3034**
4	657.4714	1.4351e+03	777.6286	1.0131e+03
5	726.8701	1.4229e+03	**696.0299**	1.0102e+03
6	650.4856	1.4447e+03	794.2144	1.0024e+03
7	706.5981	1.4490e+03	742.4019	1.0060e+03
8	709.4055	1.4204e+03	710.9945	1.0090e+03
9	717.5541	1.4430e+03	725.4459	1.0327e+03
10	696.1554	1.4498e+03	753.6446	1.0019e+03

Table 5.2 Experimental results with a 4-UAV flock

Exp.	Min length (m)	Max length (m)	Max-Min (m)	Mean length (m)
1	494.4280	1.3122e+03	817.772	810.8323
2	483.7776	1.3075e+03	823.7224	805.5217
3	479.4430	1.3086e+03	829.157	801.8152
4	526.0294	1.3277e+03	801.6706	822.4155
5	475.9247	1.2945e+03	818.5753	**797.8696**
6	484.6782	1.3132e+03	828.5218	805.2276
7	465.3765	1.2912e+03	825.8235	811.1517
8	501.3612	1.2874e+03	**786.0388**	801.8962
9	485.3590	1.3036e+03	818.241	809.5133
10	461.4236	1.2736e+03	812.1764	802.6272

Table 5.3 Experimental results with a 5-UAV flock

Exp.	Min length (m)	Max length (m)	Max-Min (m)	Mean length (m)
1	330.7004	1.1211e+03	790.3996	673.1732
2	340.9493	1.1020e+03	761.0507	663.6878
3	352.0763	1.0924e+03	**740.3237**	**659.2494**
4	323.4266	1.1190e+03	795.5734	679.3655
5	348.1068	1.0966e+03	748.4932	663.7847
6	325.4634	1.1141e+03	788.6366	668.6771
7	337.3423	1.0971e+03	759.7577	661.7655
8	317.5902	1.1129e+03	795.3098	667.6786
9	358.7191	1.1332e+03	774.4809	669.0999
10	350.2807	1.1119e+03	761.6193	662.5721

Table 5.4 Experimental results with a 6-UAV flock

Exp.	Min length (m)	Max length (m)	Max-Min (m)	Mean length (m)
1	197.8880	1.0448e+03	846.912	567.7679
2	207.6521	1.0392e+03	831.574	563.4251
3	191.4326	1.0435e+03	852.0674	562.9172
4	188.7654	1.0332e+03	844.4346	561.4877
5	205.4141	1.0501e+03	844.6859	568.3652
6	223.4688	961.4863	**738.0175**	560.6003
7	215.5982	1.0441e+03	828.5018	560.3835
8	187.3015	1.0407e+03	853.3985	**558.9720**
9	205.7313	1.0626e+03	856.8687	570.2513
10	204.2937	1.0568e+03	852.5063	562.7059

Table 5.5 Experimental results with a 7-UAV flock

Exp.	Min length (m)	Max length (m)	Max-Min (m)	Mean length (m)
1	164.6611	948.0504	783.3893	493.4981
2	152.1319	925.2858	773.1539	480.3906
3	156.5260	926.5934	770.0674	483.5770
4	165.2738	961.4037	796.1299	495.3582
5	158.6764	940.2803	781.6039	487.7900
6	156.0176	954.8158	798.7982	487.3445
7	151.8748	1.0098e+03	857.9252	495.4426
8	151.7503	936.4922	784.7419	484.7355
9	158.6935	928.4387	**769.7452**	488.1152
10	114.3742	931.9550	817.5808	**441.9657**

Table 5.6 Experimental results with a 8-UAV flock

N. of UAVs	**Mean lengths std dev. (m)**	**Max-min diff. std dev. (m)**
3	11.0335	26.8535
4	10.5408	36.9055
5	6.9641	**13.4088**
6	6.0276	20.1442
7	**3.8117**	35.2749
8	15.5383	27.1066

Table 5.7 Statistical analysis of the obtained results

- define region of interests and no-fly zones on georeferenced 2D maps or digital elevation models
- decompose the ROI via Morse-based decomposition
- perform a complete coverage path planning of the decomposed zone
- assign coverage trajectories to the UAVs composing the flock via negotiation algorithms
- obtain a waypoint list for testing on real hardware

From the experimental results, a general relationship between minimum average lengths and maximum balance in path assigned to UAVs seems to be not trivial to determine. However, although the number of experiments is not still large enough from the perspective of a full statistical relevance, the evidences reported in the previous section do represent an encouraging set of preliminary results. This calls for further investigation, with a larger number of experiments

and by considering metrics related to the geometric distributions of the flock starting formation within the ROI.

Moreover, finding an analytically optimal solution for CCPP missions with a team of aerial vehicles may imply a computational burden (and then a computational time) which would make it useless. The developed framework allows to test several different scenarios, with respect to the number of UAVs and decompositions. Eventually, the most satisfactory solution can be chosen. The main limit of this work is that vehicles' trajectories are computed offline.

Chapter 6

Discussion

In this thesis the problem of unmanned ground vehicle navigation in unstructured and heterogeneous outdoor environments has been addressed from different perspectives: autonomous navigation, informed teleoperation, and with the help of a flock of aerial vehicles.

The autonomous strategy is grounded on a geometry-based traversability analysis of the environment model. The main novelty of this work resides in the reconstruction of such model, with the support of an autonomous UAV through aerial photogrammetry. The result of this analysis is a 2D map of traversing costs, which has been used with a grid-based path planning algorithm in order to avoid hazardous and difficult zones.

The teleoperation solution consists in a novel immersive interface which leverages mixed reality to combine images acquired from the real scenario with a 3D model of the environment and graphical aids, representing data information. All these elements, combined in a single innovative interface, proved to be a valuable support for

the operators, resulting in an enhancement of the remote driving performance, as the on-field pilot study has shown.

Finally, the deployment of flocks of aerial vehicles to assist in UGV navigation has been investigated. In particular, the multi-UAV team has been adopted for the 3D reconstruction of the area of interest, through a complete coverage path planning. Such model can be used as input for the geometry-based traversability analysis cited above. A novel integrated software framework has been developed in order to manage the overall coverage mission, from ROI and no-fly zones definition up to waypoint list generation for the autopilot.

In all of the described works the adoption of three-dimensional representations of the environment, proved to be an effective approach, for both the traversability analysis and the autonomous navigation of aerial and ground vehicles in the considered applications. This approach is made viable by current state of the art in aerial photogrammetric reconstruction, which has reached remarkable results, in quality and performance. A crucial aspect is the possibility to include also the appearance of the reconstructed scene (i.e. colors, textures, and patterns), in addition to geometry, which makes the model even more realistic. Moreover, digital elevation models provided environment representations rich enough to derive an appreciable metric for traversability assessment. Thanks to their simple data structure, DEMs are efficient and fairly approximate models, which are extremely profitable for outdoor navigation of ground vehicles.

6.1 Future developments

The results achieved with the works described in the previous chapters are, of course, encouraging and motivate to pursue this research topic. However, there are still some points which call for further investigation and consideration. Focusing on the traversability analysis problem, the purely geometric approach developed is planned to be extended in order to include environment appearance as well. To this end, realistic appearance of 3D models can be really helpful for development and testing. In the literature, appearance-based approaches have been mainly focused on terrain type identification and classification. However, it would be interesting to explore the adoption of appearance information with *robot learning* solutions, to directly infer traversing costs. Such solutions offer some advantages. First, the independence from the environment representation considered, thus allowing to work with overhanging structures or tunnels. Secondly, the independence from the vehicle specificity, since the approach can be interchangeably applied to legged, wheeled, and tracked vehicles. Some steps in the research activity have already been done in this sense; however, results were not mature enough to be included in this thesis. The main drawback of robot learning approaches is the rather large amount of data needed to achieve appreciable results during the training phase, especially in the case of deep neural networks. Furthermore, depending on the application, it could not be simple or feasible to collect a large number of training examples.

Although the final solution is still far to be defined, especially in very challenging environments, it can be claimed that the problem

of autonomous navigation in unstructured environments will gain an even increased interest from robotics research in the near future, since it will be presumably the next goal of self-driving car research and industry.

Bibliography

[1] Abbeel, P. and Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*. ACM.

[2] Acar, E. U., Choset, H., Rizzi, A. A., Atkar, P. N., and Hull, D. (2002). Morse decompositions for coverage tasks. *The international journal of robotics research*, 21(4):331–344.

[3] Andersen, J. C., Blas, M. R., Ravn, O., Andersen, N. A., and Blanke, M. (2006). Traversable terrain classification for outdoor autonomous robots using single 2D laser scans. *Integrated Computer-aided engineering*, 13(3):223–232.

[4] ardupilot (2018). The ardupilot project. [online] http://ardupilot.org/. Accessed Nov. 2018.

[5] Atkar, P. N., Greenfield, A., Conner, D. C., Choset, H., and Rizzi, A. A. (2005). Uniform coverage of automotive surface patches. *The International Journal of Robotics Research*, 24(11):883–898.

[6] Balta, H., De Cubber, G., Doroftei, D., Baudoin, Y., and Sahli, H. (2013). Terrain traversability analysis for off-road robots using time-of-flight 3D sensing. In *7th IARP International Workshop on Robotics for Risky Environment-Extreme Robotics, Saint-Petersburg, Russia*.

[7] Barrientos, A., Colorado, J., del Cerro, J., Martinez, A., Rossi, C., Sanz, D., and Valente, J. (2011). Aerial remote sensing in agricul-

ture: A practical approach to area coverage and path planning for fleets of mini aerial robots. *Journal of Field Robotics*, 28(5):667–689.

[8] Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., and Karol, Z. (2016). End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*.

[9] Bonaccorso, F., Longo, D., and Muscato, G. (2012). The U-Go robot. In *World Automation Congress*, pages 1–6.

[10] Bonnafous, D., Lacroix, S., and Siméon, T. (2001). Motion generation for a rover on rough terrains. In *2001 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 784–789. IEEE.

[11] Braun, T., Bitsch, H., and Berns, K. (2008). Visual terrain traversability estimation using a combined slope/elevation model. In *Annual Conference on Artificial Intelligence*, pages 177–184. Springer.

[12] Burigat, S. and Chittaro, L. (2007). Navigation in 3D virtual environments: Effects of user experience and location-pointing navigation aids. *International Journal of Human-Computer Studies*, 65(11):945–958.

[13] Calhoun, G. L., Draper, M. H., Abernathy, M. F., Patzek, M., and Delgado, F. (2005). Synthetic vision system for improving unmanned aerial vehicle operator situation awareness. In *Enhanced and Synthetic Vision 2005*, volume 5802, pages 219–231. International Society for Optics and Photonics.

[14] Cantelli, L., Guastella, D., Melita, C., and Muscato, G. (2017a). UGV navigation in unstructured environments through UAV survey. In *Human-centric Robotics-Proceedings Of The 20th International Conference Clawar 2017*, page 327. World Scientific.

[15] Cantelli, L., Guastella, D. C., Longo, D., Melita, C. D., and Muscato, G. (2018). Coverage path planning by swarm of UAVs for UGV traversability analysis. 21st International Symposium on Measurement and Control in Robotics (ISMCR '18).

[16] Cantelli, L., Laudani, P., Melita, C., and Muscato, G. (2017b). UAV/UGV cooperation to improve navigation capabilities of a mobile robot in unstructured environments. In *Advances in Cooperative Robotics*, pages 217–224. World Scientific.

[17] Cao, Z. L., Huang, Y., and Hall, E. L. (1988). Region filling operations with random obstacle avoidance for mobile robots. *Journal of Robotic systems*, 5(2):87–102.

[18] Cheng, P., Keller, J., and Kumar, V. (2008). Time-optimal UAV trajectory planning for 3D urban structure coverage. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2750–2757.

[19] Chintamani, K., Cao, A., Ellis, R. D., and Pandya, A. K. (2010). Improved telemanipulator navigation during display-control misalignments using augmented reality cues. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 40(1):29–39.

[20] Choset, H. M., Hutchinson, S., Lynch, K. M., Kantor, G., Burgard, W., Kavraki, L. E., and Thrun, S. (2005). *Principles of robot motion: theory, algorithms, and implementation*. MIT press.

[21] Di Franco, C. and Buttazzo, G. (2016). Coverage path planning for UAVs photogrammetry with energy and resolution constraints. *Journal of Intelligent & Robotic Systems*, 83(3-4):445–462.

[22] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271.

[23] Dubbelman, G., van der Mark, W., van den Heuvel, J. C., and Groen, F. C. (2007). Obstacle detection during day and night conditions using stereo vision. In *2007 IEEE/RSJ International Con-*

ference on Intelligent Robots and Systems (IROS), pages 109–116. IEEE.

[24] Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, (6):46–57.

[25] Endsley, M. R. (1988). Design and evaluation for situation awareness enhancement. In *Proceedings of the Human Factors Society annual meeting*, volume 32, pages 97–101. SAGE Publications Sage CA: Los Angeles, CA.

[26] Englot, B. and Hover, F. S. (2013). Three-dimensional coverage planning for an underwater inspection robot. *The International Journal of Robotics Research*, 32(9-10):1048–1073.

[27] Fedorenko, R., Gabdullin, A., and Fedorenko, A. (2018). Global UGV path planning on point cloud maps created by UAV. In *2018 3rd IEEE International Conference on Intelligent Transportation Engineering (ICITE)*, pages 253–258. IEEE.

[28] Ferland, F., Pomerleau, F., Le Dinh, C. T., and Michaud, F. (2009). Egocentric and exocentric teleoperation interface using real-time, 3D video projection. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 37–44. ACM.

[29] Fong, T. and Thorpe, C. (2001). Vehicle teleoperation interfaces. *Autonomous robots*, 11(1):9–18.

[30] Galceran, E. and Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous systems*, 61(12):1258–1276.

[31] Gibson, J. J. (2014). *The ecological approach to visual perception: classic edition*. Psychology Press.

[32] Grocholsky, B., Keller, J., Kumar, R. V., and Pappas, G. J. (2006). Cooperative air and ground surveillance. *IEEE Robotics Automation Magazine*, 13(3):16–25.

[33] Guastella, D. C., Cantelli, L., Melita, C. D., and Muscato, G. (2017). A global path planning strategy for a UGV from aerial elevation maps for disaster response. In *ICAART (1)*, pages 335–342.

[34] Guastella, D. C., Cavallaro, N. D., Melita, C. D., Savasta, M., and Muscato, G. (2018). 3D path planning for UAV swarm missions. In *Proceedings of the 2018 2nd International Conference on Mechatronics Systems and Control Engineering*, pages 33–37. ACM.

[35] Happold, M., Ollis, M., and Johnson, N. (2006). Enhancing supervised terrain classification with predictive unsupervised learning. In *Robotics: science and systems*. Citeseer.

[36] Harrison, B. L., Kurtenbach, G., and Vicente, K. J. (1995). An experimental evaluation of transparent user interface tools and information content. In *Proceedings of the 8th annual ACM symposium on User interface and software technology*, pages 81–90. ACM.

[37] Hart, P. E., Nilsson, N. J., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107.

[38] Hirose, N., Sadeghian, A., Vázquez, M., Goebel, P., and Savarese, S. (2018). GONet: A semi-supervised deep learning approach for traversability estimation. *arXiv preprint arXiv:1803.03254*.

[39] Joho, D., Stachniss, C., Pfaff, P., and Burgard, W. (2007). Autonomous exploration for 3D map learning. In *Autonome Mobile Systeme 2007*, pages 22–28. Springer.

[40] Kahn, G., Villaflor, A., Ding, B., Abbeel, P., and Levine, S. (2018). Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8.

[41] Kelly, A., Chan, N., Herman, H., Huber, D., Meyers, R., Rander, P., Warner, R., Ziglar, J., and Capstick, E. (2011). Real-time photorealistic virtualized reality interface for remote mobile robot control. *The International Journal of Robotics Research*, 30(3):384–404.

[42] Keyes, B., Casey, R., Yanco, H. A., Maxwell, B. A., and Georgiev, Y. (2006). Camera placement and multi-camera fusion for remote robot operation. In *Proceedings of the IEEE international workshop on Safety, Security and Rescue Robotics*, pages 22–24.

[43] Khan, A., Noreen, I., and Habib, Z. (2017). On complete coverage path planning algorithms for non-holonomic mobile robots: Survey and challenges. *Journal of Information Science and Engineering*.

[44] Kim, D., Oh, S. M., and Rehg, J. M. (2007). Traversability classification for UGV navigation: A comparison of patch and superpixel representations. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3166–3173.

[45] Krückel, K., Nolden, F., Ferrein, A., and Scholl, I. (2015). Intuitive visual teleoperation for UGVs using free-look augmented reality displays. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4412–4417.

[46] Kubota, T., Kuroda, Y., Kunii, Y., and Yoshimitsu, T. (2001). Path planning for newly developed microrover. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3710–3715.

[47] Li, Y., Chen, H., Er, M. J., and Wang, X. (2011). Coverage path planning for UAVs based on enhanced exact cellular decomposition method. *Mechatronics*, 21(5):876–885.

[48] Lin, Q. and Kuo, C. (1997). Virtual tele-operation of underwater robots. In *1997 IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1022–1027.

[49] Livatino, S., Banno, F., and Muscato, G. (2012). 3-D integration of robot vision and laser data with semiautomatic calibration in augmented reality stereoscopic visual interface. *IEEE Transactions on Industrial Informatics*, 8(1):69–77.

[50] Livatino, S., Muscato, G., and Privitera, F. (2009). Stereo viewing and virtual reality technologies in mobile robot teleguide. *IEEE Transactions on Robotics*, 25(6):1343–1355.

[51] Lu, L., Ordonez, C., Collins Jr, E. G., Coyle, E., and Palejiya, D. (2011). Terrain surface classification with a control mode update rule using a 2D laser stripe-based structured light sensor. *Robotics and Autonomous Systems*, 59(11):954–965.

[52] Luettel, T., Himmelsbach, M., and Wuensche, H.-J. (2012). Autonomous ground vehicles-concepts and a path to the future. *Proceedings of the IEEE*, 100(Centennial-Issue):1831–1839.

[53] Marin, R., Sanz, P. J., and Sanchez, J. (2002). A very high level interface to teleoperate a robot via web including augmented reality. In *2002 IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2725–2730.

[54] Martin, S. C. (2018). *Proprioceptive sensing of traversability for long-term navigation of robots*. PhD thesis, Queensland University of Technology.

[55] Martins, H. and Ventura, R. (2009). Immersive 3-D teleoperation of a search and rescue robot using a head-mounted display. In *2009 IEEE Conference on Emerging Technologies & Factory Automation (ETFA)*, pages 1–8.

[56] Milgram, P. and Kishino, F. (1994). A taxonomy of mixed reality visual displays. *IEICE Transactions on Information and Systems*, 77(12):1321–1329.

[57] Milgram, P., Takemura, H., Utsumi, A., and Kishino, F. (1995). Augmented reality: A class of displays on the reality-virtuality

continuum. In *Telemanipulator and telepresence technologies*, volume 2351, pages 282–293.

[58] Miller, D. P. and Machulis, K. (2005). Visual aids for lunar rover tele-operation. In *proceedings of 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space*.

[59] Monferrer, A. and Bonyuet, D. (2002). Cooperative robot teleoperation through virtual reality interfaces. In *Sixth International Conference on Information Visualisation*, pages 243–248. IEEE.

[60] Moon, S., Oh, E., and Shim, D. H. (2013). An integral framework of task assignment and path planning for multiple unmanned aerial vehicles in dynamic environments. *Journal of Intelligent & Robotic Systems*, 70(1-4):303–313.

[61] Nguyen, L. A., Bualat, M., Edwards, L. J., Flueckiger, L., Neveu, C., Schwehr, K., Wagner, M. D., and Zbinden, E. (2001). Virtual reality interfaces for visualization and control of remote vehicles. *Autonomous Robots*, 11(1):59–68.

[62] Nguyen, T. T. H., Duval, T., and Fleury, C. (2013). Guiding techniques for collaborative exploration in multi-scale shared virtual environments. In *International Conference on Computer Graphics Theory and Applications GRAPP*, pages 327–336.

[63] Nielsen, C. W., Goodrich, M. A., and Ricks, R. W. (2007). Ecological interfaces for improving mobile robot teleoperation. *IEEE Transactions on Robotics*, 23(5):927–941.

[64] Noguchi, N., Reid, J. F., Zhang, Q., Will, J. D., and Ishii, K. (1998). Development of robot tractor based on RTK-GPS and gyroscope. In *ASAE Annual Meeting*.

[65] Norouzi, M., Miro, J. V., and Dissanayake, G. (2012). Planning high-visibility stable paths for reconfigurable robots on uneven terrain. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2844–2849. IEEE.

[66] nvidiatx1 (2018). Jetson TX1 developer kit by NVidia. [online] https://www.nvidia.com/en-us/autonomous-machines/embedded-systems-dev-kits-modules/. Accessed Nov. 2018.

[67] oculus (2018). Oculus Rift. [online] https://www.oculus.com/rift/. Accessed Nov. 2018.

[68] Ohno, K., Tsubouchi, T., Shigematsu, B., Maeyama, S., and Yuta, S. (2003). Outdoor navigation of a mobile robot between buildings based on DGPS and odometry data fusion. In *2003 IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1978–1984. IEEE.

[69] Okamura, A. M. (2004). Methods for haptic feedback in teleoperated robot-assisted surgery. *Industrial Robot: An International Journal*, 31(6):499–508.

[70] Okura, F., Ueda, Y., Sato, T., and Yokoya, N. (2013). Teleoperation of mobile robots by generating augmented free-viewpoint images. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 665–671. IEEE.

[71] Ott, R., Gutiérrez, M., Thalmann, D., and Vexo, F. (2006). Advanced virtual reality technologies for surveillance and security applications. In *Proceedings of the 2006 International Conference on Virtual Reality continuum and its applications*, pages 163–170. ACM.

[72] Papadakis, P. (2013). Terrain traversability analysis methods for unmanned ground vehicles: A survey. *Engineering Applications of Artificial Intelligence*, 26(4):1373–1385.

[73] Papadakis, P. and Pirri, F. (2012). 3D mobility learning and regression of articulated, tracked robotic vehicles by physics-based optimization. In *Virtual Reality Interaction and Physical Simulation, Eurographics*.

[74] Perez-Imaz, H. I., Rezeck, P. A., Macharet, D. G., and Campos, M. F. (2016). Multi-robot 3D coverage path planning for first responders teams. In *2016 IEEE International Conference on Automation Science and Engineering (CASE)*, pages 1374–1379. IEEE.

[75] Piguet, L., Hine, B., Hontalas, P., Fong, T., and Nygren, E. (1996). The Virtual Environment Vehicle Interface: a dynamic, distributed and flexible virtual environment. In *IMAGINA '96: New Frontiers of CyberExistence*. Citeseer.

[76] pix4d (2018). Professional photogrammetry software for drone-based mapping. [online] https://pix4d.com/product/pix4dmapper-photogrammetry-software/. Accessed Nov. 2018.

[77] Preusche, C., Ortmaier, T., and Hirzinger, G. (2002). Teleoperation concepts in minimal invasive surgery. *Control engineering practice*, 10(11):1245–1250.

[78] Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). ROS: an open-source Robot Operating System. In *ICRA workshop on open source software*, volume 3.

[79] Quintero, C. P., Fomena, R. T., Shademan, A., Wolleb, N., Dick, T., and Jägersand, M. (2013). SEPO: Selecting by pointing as an intuitive human-robot command interface. In *2013 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1166–1171.

[80] Rusu, R. (2009). *Semantic 3D object maps for everyday manipulation in human living environments*. PhD thesis, Technische Universität Müchen.

[81] Santamaria-Navarro, À., Teniente, E. H., Morta, M., and Andrade-Cetto, J. (2015). Terrain classification in complex three-dimensional outdoor environments. *Journal of Field Robotics*, 32(1):42–60.

[82] Schwarz, M., Rodehutskors, T., Droeschel, D., Beul, M., Schreiber, M., Araslanov, N., Ivanov, I., Lenz, C., Razlaw, J., Schüller, S., Schwarz, D., Topalidou-Kyniazopoulou, A., and Behnke, S. (2017). NimbRo rescue: Solving disaster-response tasks with the mobile manipulation robot momaro. *Journal of Field Robotics*, 34(2):400–425.

[83] Siegwart, R., Nourbakhsh, I. R., Scaramuzza, D., and Arkin, R. C. (2011). *Introduction to autonomous mobile robots*. MIT press.

[84] Silver, D., Bagnell, J. A., and Stentz, A. (2010). Learning from demonstration for autonomous navigation in complex unstructured terrain. *The International Journal of Robotics Research*, 29(12):1565–1592.

[85] Silver, D., Sofman, B., Vandapel, N., Bagnell, J. A., and Stentz, A. (2006). Experimental analysis of overhead data processing to support long range navigation. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2443–2450. IEEE.

[86] Singh, S., Simmons, R., Smith, T., Stentz, A., Verma, V., Yahja, A., and Schwehr, K. (2000). Recent progress in local and global traversability for planetary rovers. In *2000 IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1194–1200. IEEE.

[87] Stentz, A. (1994). Optimal and efficient path planning for partially-known environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 94, pages 3310–3317.

[88] Stoker, C. R., Zbinden, E., Blackmon, T. T., Kanefsky, B., Hagen, J., Neveu, C., Rasmussen, D., Schwehr, K., and Sims, M. (1999). Analyzing pathfinder data using virtual reality and superresolved imaging. *Journal of Geophysical Research: Planets*, 104(E4):8889–8906.

[89] Stückler, J., Schwarz, M., Schadler, M., Topalidou-Kyniazopoulou, A., and Behnke, S. (2016). NimbRo Explorer:

Semiautonomous exploration and mobile manipulation in rough terrain. *Journal of Field Robotics*, 33(4):411–430.

[90] Suzuki, M., Terada, E., Saitoh, T., and Kuroda, Y. (2010). Vision based far-range perception and traversability analysis using predictive probability of terrain classification. In *2010 41st International Symposium on Robotics (ISR) and 6th German Conference on Robotics (ROBOTIK)*, pages 1–6.

[91] Taylor, Z., Nieto, J., and Johnson, D. (2015). Multi-modal sensor calibration using a gradient orientation measure. *Journal of Field Robotics*, 32(5):675–695.

[92] Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic robotics*. MIT press.

[93] Torres, M., Pelta, D. A., Verdegay, J. L., and Torres, J. C. (2016). Coverage path planning with unmanned aerial vehicles for 3D terrain reconstruction. *Expert Systems with Applications*, 55:441–451.

[94] unity3d (2018). Unity3D game engine. [online] https://unity3d.com/. Accessed Nov. 2018.

[95] Vandapel, N., Donamukkala, R. R., and Hebert, M. (2006). Unmanned ground vehicle navigation using aerial ladar data. *The International Journal of Robotics Research*, 25(1):31–51.

[96] verve (2018). NASA Verve: Interactive 3D visualization within Eclipse. [online] https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20140013445.pdf. Accessed Nov. 2018.

[97] Vincent, P. and Rubin, I. (2004). A framework and analysis for cooperative search using UAV swarms. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 79–86. ACM.

[98] Wei, G., Hus, D., Lee, W. S., Shen, S., and Subramanian, K. (2017). Intention-Net: Integrating planning and deep learning for goal-directed autonomous navigation. *arXiv preprint arXiv:1710.05627*.

[99] Wettergreen, D., Bualat, M., Christian, D., Schwehr, K., Thomas, H., Tucker, D., and Zbinden, E. (1998). Operating nomad during the Atacama Desert trek. In *Field and Service Robotics*, pages 82–89. Springer.

[100] Wu, H., Liu, B., Su, W., Chen, Z., Zhang, W., Ren, X., and Sun, J. (2017). Optimum pipeline for visual terrain classification using improved bag of visual words and fusion methods. *Journal of Sensors*, 2017.

[101] Wulfmeier, M., Rao, D., Wang, D. Z., Ondruska, P., and Posner, I. (2017). Large-scale cost function learning for path planning using deep inverse reinforcement learning. *The International Journal of Robotics Research*, 36(10):1073–1087.

[102] Yamada, H., Tao, N., and DingXuan, Z. (2008). Construction tele-robot system with virtual reality. In *2008 IEEE Conference on Robotics, Automation and Mechatronics*, pages 36–40. IEEE.

[103] Yanco, H. A. and Drury, J. L. (2007). Rescuing interfaces: A multi-year study of human-robot interaction at the AAAI robot rescue competition. *Autonomous Robots*, 22(4):333–352.

[104] Ye, C. (2007). Navigating a mobile robot by a traversability field histogram. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(2):361–372.

[105] Zalud, L. (2006). ARGOS-system for heterogeneous mobile robot teleoperation. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 211–216. IEEE.

[106] zedcam (2018). ZED 2K stereo camera by Stereolabs. [online] https://www.stereolabs.com/zed/. Accessed Nov. 2018.