A ROS Gazebo plugin to simulate ARVA sensors

A ROS Gazebo plugin to simulate ARVA sensors

Jonathan Cacace[1], Nicola Mimmo[2] and Lorenzo Marconi[2]

***Abstract*— This paper addresses the problem to simulate ARVA sensors using ROS and Gazebo. ARVA is a French acronym which stands for *Appareil de Recherche de Victims en Avalanche* and represents the forefront technology adopted in Search & Rescue operations to localize victims of avalanches buried under the snow. The aim of this paper is to describe the mathematical and theoretical background of the transceiver, discussing its implementation and integration with ROS allowing researchers to develop faster and smarter Search &Rescue strategies based on ARVA receiver data. To assess the effectiveness of the proposed sensor model, We present a simulation scenario in which an Unmanned Aerial Vehicle equipped with the transceiver sensor performs a basic S&R pattern using the output of ARVA system.**

I. Introduction

In the future, Search & Rescue (S&R) missions will benefit from the use of fast and smart mobile robots. Indeed, the effectiveness of robots is twofold: from one hand, they can support rescuers to improve the efficiency of search operations and, on the other hand, they can even substitute humans to decrease the risks associated to operations in harsh environments. To this end, several ground [1] [2] [3] and aerial [4] [5] robotic platforms characterized by different capabilities and equipment have been developed to support S&R operations in different scenarios. In this context, an important challenge in S&R robotics is the one addressed by the European projects *SHERPA* [6] and *AIRBorne*[1] which aim to provide a feasible and effective robotic solution to support alpine S&R teams in rescuing avalanche victims. Unfortunately, the peculiarities of this S&R domain make practically impossible the usage of common technologies like phones and ultra-wide band localization systems. For this reason a special system, called ARVA[2], is currently used and it represents one of the forefront technologies nowadays used in S&R operations in case avalanches [7].

The ARVA devices consist of two main elements: a *transmitter* and a *receiver*, which are operated alternatively. Excursionists and skiers who use the ARVA normally set it in the transmitting mode so that, in the accidental case of avalanche, the system is already in the right operative mode. On the other hand, rescuers operate their ARVA devices in the receiving mode to be able to detect the transmitter' signal. In the receiver mode, the ARVA devices provide information about the electromagnetic field, emitted by the transmitter, which is exploited to guide the rescuer toward the victim. This information is usually made available to the rescuers in terms of magnetic distance and magnetic direction to the victim. The rescue team navigates the avalanche site by looking for a valid ARVA signal, with a formation and a strategy that depend on several aspects such as the number of rescuers, the kind of avalanche and the number of the missing people. When a valid ARVA signal has been detected, rescuers start following the electromagnetic flux lines emitted by the ARVA transmitter, and visualized on the screen of the ARVA receivers.

Despite the huge amount of sensors supported by ROS simulation tools [8], a method to simulate the ARVA transceiver has not yet been developed. Then, the aim of this work is twofold and consists in presenting the `arva_sim`, a ROS package implementing the ARVA system, and in interfacing it with the Gazebo simulator. This package can be found at this link: `https://github.com/jocacace/arva_sim`. Installing the `arva_sim` package, a new couple of plugins will be available in the Gazebo simulations. The main goal of the `arva_sim` package is to allow researchers and developers to equip simulated robots with ARVA devices, in order to test and benchmark efficient and autonomous search strategies. Moreover, Unmanned Aerial Vehicles (UAVs) are extensively employed in S&R operations in several domains [9] [10] [11], with a particular focus on the alpine scenarios, [12] [13] [14] demonstrate how S&R operations can benefit from the use of autonomous aerial robots to survey the environment and collect evidences about the position of a missing person. For this reason, this work also proposes a S&R test case in agreement with the guidelines of the European project *AIRBorne*. In detail, a simulated UAV equipped with the proposed ARVA sensor is used to autonomously perform the standard human made search operations.

The reminder of the work is organized as follows. In Section II the theoretical background of the ARVA system is presented, while in Section III the description of the software architecture of the `arva_sim` package is presented. Finally, Section IV proposes a case study to assess the effectiveness of the ARVA simulator. We additionally included a brief tutorial to download, install and run a first test case of the ARVA simulator in the Appendix of this work.

This research was supported by the European Project AIRBORNE, Grant Agreement no. 780960, web site `https://www.airborne-project.eu/`.

[1]Jonathan Cacace is with the University of Naples "Federico II", Department of Electrical Engineering and Information Technology. Via Claudio, 21 - 80125 - Naples (NA) - Italy `jonathan.cacace@unina.it`

[2]Nicola Mimmo and Lorenzo Marconi are with the are with the Department of Electrical, Electronic and Information Engineering, University of Bologna, Italy, Viale del Risorgimento 2, 40136 - Bologna (BO)`{nicola.mimmo2, lorenzo.marconi}@unibo.it`

[1] `https://www.airborne-project.eu/`

[2]The Appareil de Recherche de Victimes

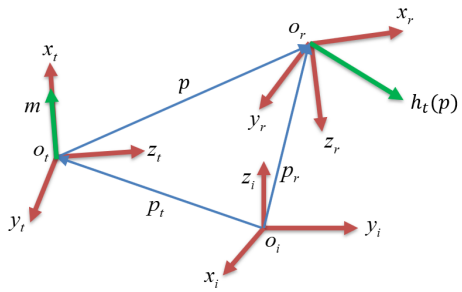

Fig. 1: Reference frames definition

II. Background

A. Notation

Three reference systems are used to derive the mathematical model. The inertial reference frame is indicated by $\mathcal{F}_i(O_i, x_i, y_i, z_i)$, the second reference frame is attached to the ARVA transmitter and it is denoted by $\mathcal{F}_t(O_t, x_t, y_t, z_t)$, with x_t, y_t and z_t aligned with the three antennae constituting the device. Finally, the third reference frame, $\mathcal{F}_r(O_r, x_r, y_r, z_r)$, centered at O_r is attached to the receiver. The relative rotation from $\mathcal{F}_r$ to $\mathcal{F}_i$ is indicated by the matrix ${}^iR_r \in SO(3)$ whereas the matrix ${}^iR_t \in SO(3)$ rotates a vector described in $\mathcal{F}_t$ to $\mathcal{F}_i$. The $SO(3)$ generator is the matrix $S(\cdot)$ which indicates a skew symmetric matrix.

The positions of the transmitter and the receiver are indicated by ${}^\#p_t$ and ${}^\#p_r$ with the superscript $\# \in \{i, r, t\}$ indicating the reference frame of description.

Finally, a subset $\mathbf{x} \subset \{x, y, z\}$ of coordinates, with cardinality indicated by $n = |\mathbf{x}|$, is defined. Then, the operator which projects the vectors in $\mathbb{R}^3$ onto the subspace of coordinates $\mathbf{x}$ is indicated by $L_{\mathbf{x}} : \mathbb{R}^3 \mapsto \mathbb{R}^n$.

B. The ARVA system

This paragraph goes through the main physical principles of the sensor and its main features that are instrumental to the development of the automatic search algorithm presented in Section IV.

1) ARVA in the transmitter mode: The ARVA transmitter generates an electromagnetic dipole, indicated by the vector m (see Figure 1) and aligned with the x_t axis of $\mathcal{F}_t$, whose associated vector field is indicated by h. Denoting by (x, y, z) the components of the vector ${}^tp_r$ expressed in $\mathcal{F}_t$ (i.e. ${}^tp_r = \text{col}(x, y, z)$), it turns out that a mathematical model of the magnetic vector field is given by (see [15])

$$ {}^th({}^tp_r) = \frac{\|m\|}{4\pi\|{}^tp_r\|^5} \mathrm{A}({}^tp_r) \tag{1} $$

where

$$ \mathrm{A}({}^tp_r) := \begin{bmatrix} 2x^2 - y^2 - z^2 \\ 3xy \\ 3xz \end{bmatrix} $$

The flux lines described by (1) are depicted in Figure 2 (red lines). Finally, the nominal electromagnetic field ${}^ih_0(\cdot)$, expressed in the inertial coordinate frame, is obtained by a roto-translation of the field ${}^th(\cdot)$ by the matrix ${}^iR_t$:

$$ {}^ih_0 = {}^iR_t\, {}^th\left({}^tR_i({}^ip_r - {}^ip_t)\right). \tag{2} $$

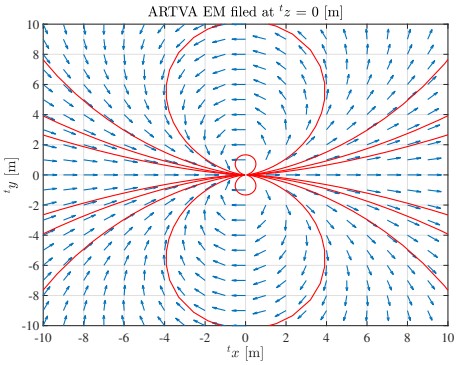

Fig. 2: Flux lines and iso-power lines for a theoretical magnetic dipole.

2) ARVA in the receiver mode: The receiver is an electromagnetic sensor with three antennas aligned along the axes x_r, y_r and z_r. The magnetic field read by the receiver, ${}^rh$, is given by the projection of the inertial vector ${}^ih_0$ onto the $\mathcal{F}_r$ frame:

$$ {}^rh := {}^rR_i\, {}^ih_0 + Q\, {}^rR_i\, {}^rw = {}^rR_i \frac{\kappa\|m\|}{4\pi\|{}^tp_r\|^5} {}^iR_t\, A({}^tp_r) + Q\, {}^rR_i\, {}^iw \tag{3} $$

where ${}^iw$ indicates the noise due to ElectroMagnetic Interference (EMI) expressed in the inertial frame whereas the matrix $Q = \text{diag}(q_1, q_2, q_3)$, $q_i \in (0, 1]$ for $i = 1, 2, 3$, indicates the noise sensitivity of the i-th antenna.

C. ElectroMagnetic Interference

A field test campaign was performed by measuring the relative position of the receiver with respect to the transmitter (by means of GPS receivers), their relative attitude (by means of Inertial Measurement Units) and the ARVA data. The outcome of this test campaign is that, if the distance between the receiver and the transmitter is larger that approximately 60 meters, the ARVA outputs are identically equal to -1. This result is reported in Figure 3. A reverse engineering process allowed for the determination of the noise floor which is described by the equations 4 and 5. The maximum

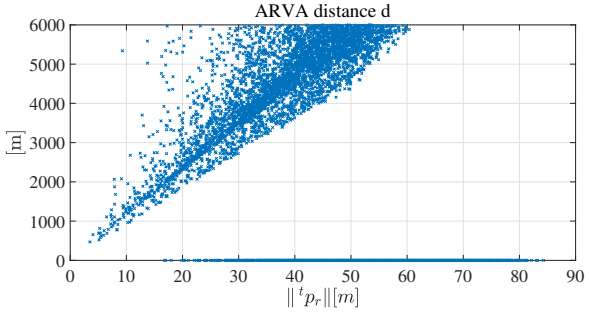

Fig. 3: The distance d provided by commercial ARVAs: the points represent the data d, determined in presence of the environment noise, over the actual distance $\|{}^tp_r\|$.

environment induced EMI are estimated to be equivalent to an ARVA signal emitted by a transmitter approximately

located 80 meters far from the receiver. Starting from these data an equivalent noise intensity is determined by imposing:

$$\max \|{}^{r}h\| = \max \|Q^{r}\| \|{}^{i}w\| = \frac{\kappa \|m\|}{2\pi \|{}^{t}p_{r_{en}}\|^3} \tag{4}$$

with $\|{}^{t}p_{r_{en}}\| = 80$. The norm $\|{}^{i}w\|$ solving the equality of Eq. (4), namely $\|{}^{i}w_{en}\|$, is obtained by the following expression:

$$\|{}^{i}w_{en}\| = \frac{\kappa \|m\|}{2\pi \max \|Q^{r}\| \|{}^{t}p_{r_{en}}\|^3}. \tag{5}$$

D. Commercial devices

The factor shape of the ARVA devices available from the market, characterized by a parallelepiped shape specifically studied to improve the transportability and werearability, imposes that the three antennas are necessarily characterized by different ferromagnetic core geometries with, then, different efficiency in reading the magnetic field intensity. In particular, the antenna placed along x_r has the biggest ferromagnetic core and acts as primary antenna with the best Signal to Noise Ratio (SNR). It can be efficiently used also far from the transmitter, where the magnetic field intensity becomes small with respect to the environment noises. The remaining two antennas, placed along y_r and z_r, have decreasing efficiency and act as secondary and thirdly antennas. They can be exploited in processing the output when the magnetic field is sufficiently strong, *i.e* when the receiver is sufficiently close to the transmitter, according to a strategy explained in the following. The commercial ARVA receivers yield two outputs that are denoted by d and δ, which respectively indicate the estimated distance to the transmitter (namely a proxy of $\|{}^{r}p_t\|$), and the orientation of ${}^{r}h$ on the plane $x_r - y_r$. The sensor outputs, of course, are strongly affected by the distance to the victim and by the fact that not all the three components of the magnetic field are measured with the same efficiency (due to the different SNRs). In this respect, the ARVA receivers implement a logic which exploits the measured signal ${}^{r}h$ to cluster the search area in three zones, which are indicative of the distance to the victim ("very far", "close", "very close"), and to compute the outputs $(d, \delta) \in \mathbb{Z}^2$ according to the actual zone. More specifically:

- *outer zone*: characterized by a SNR under a minimum threshold, indicative of a large distance to the victim (typically $\|{}^{t}p_r\| > 60\,\text{m}$ according to the data-sheet of the manufacturer). In this zone the signal ${}^{r}h$ is too weak to yield a reliable information and the receiver output is set to $(d, \delta) = (-1, -1)$;
- *mid zone*: if the projection of the measured signal on the receiver $x_r - y_r$ axes, namely $L^{r}_{xy}h$, is larger than a certain threshold reliable data d and δ can be computed. More specifically, the distance d is computed by exploiting the relation (1) considering only the first two components of ${}^{r}h$ as relevant signal:

$$d = \left(\frac{\kappa_{eq}}{4\pi \|L_{xy}\, {}^{r}h\|} \right)^{\frac{1}{3}} \tag{6}$$

where κ_{eq} is constant suitably tuned by the manufacturers and L_{xy} represents the projector operator on the $x_r - y_r$ axes. The distance d provided in this phase, being determined exploiting only the most sensitive antennae (over the three available), is a function of both the receiver and transmitter orientation, ${}^{i}R_r$ and ${}^{i}R_t$, and the actual relative distance, ${}^{t}p_r$, *i.e.* $d = d\left({}^{i}\mathrm{R}_r, {}^{i}\mathrm{R}_t, {}^{t}p_r\right)$. When in this operating mode the distance $d \in [300, 5000]\ cm$. On the other hand, the angle δ is determined by implementing the following equation:

$$\delta = \tan^{-1}\left(\frac{L_y\, {}^{r}h}{L_x\, {}^{r}h} \right) \tag{7}$$

where $L_x\, {}^{r}h$ and $L_y\, {}^{r}h$ indicate the projection of ${}^{r}h$ on the x_r and y_r respectively. Finally, the angle $\delta \in (-90, 90)\ deg$. It is worth nothing that, if the receiver orientation ${}^{i}R_r$ is such that $L_{xy}\, {}^{r}h = 0$, the output is $(d, \delta) = (-1, -1)$;
- *inner zone*: in this zone the strength of the signal ${}^{r}h$ is sufficiently high to allow the exploitation of all the three antennas. In the last search phase, the receiver is sufficiently close to the transmitter to exploit the third antenna to estimate, besides the first two, the distance to the victim. In this phase the distance d is not more a function of receiver and transmitter relative orientation, *i.e.* $d = d\left({}^{t}p_r\right)$ while the angle δ is not more provided and set to zero. Finally, in the third search phase $d \in [0, 300)\ cm$ and $\delta = 0\ deg$ where

$$d = \left(\frac{\kappa_{eq}}{4\pi \|{}^{r}h\|} \right)^{\frac{1}{3}}. \tag{8}$$

Finally, the effect of the environment noise on the the data d provided by the commercial ARVA devices is shown in Figure 3. In detail, for distances $\|{}^{t}p_r\| > 60$ m, the strength of the electromagnetic field h_0 is comparable or below the noise power so that the sensed field ${}^{r}h$ is practically representative of the noise. Moreover, it is worth nothing that, since the clusters are based on the projection $L_{xy}\, {}^{r}h$ and thus of the receiver orientation, the distance d could be equal to -1 even if the distance $\|{}^{t}p_r\| \leq 60$ m.

III. arva_sim Ros Package

The proposed Gazebo plugins replicate the behavior of the ARVA transceiver depicted in Figure 2. For this reason, two different plugins are deployed: the receiver and transmitter. The working principle of such plugins is reported in

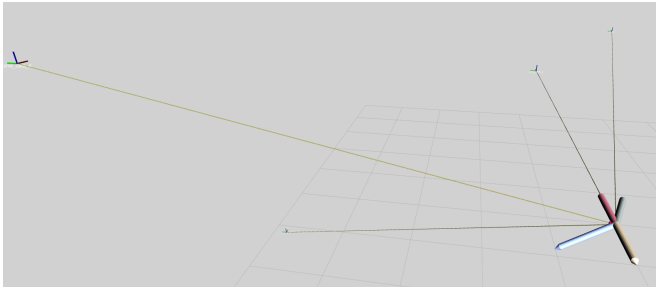

Fig. 4: The ARVA receiver detecting four different transmitters at the same time.

Figure 5, in which an ARVA transmitter is placed in the origin of the simulation scene while ROS visualization tool (RViz) [16] is used to show the direction of the flux for each position of the workspace. The software architecture for this package is reported in Figure 6 and it is detailed hereafter. Each *ARVA Transmitter* imported in the simulation scene informs the ROS system about its current pose inside the Gazebo world, while the *Receiver* collects all the ARVA poses and calculates the magnetic flux direction for each of them. The transmitter output consists on a ROS message containing the direction to reach a given transmitter and the distance from it. In order to setup the ARVA system, it is necessary to attach the devices to physical objects and spawn them in the simulation environment. The common way to do this in the ROS framework is by using *Universal Robotic Description Format* (URDF) [17] or `xacro` (XML Macros)[3] files, like already made in the example files included in the `urdf` folder of the `arva_sim` package. The main parameters characterizing the ARVA transmitters are the `<id>` number, a non-negative number used to identify a specific transmitter, the `<frame_id>`, the point of reference for ARVA data and its initial position and orientation in the simulation. These parameters must be specified in the `launch` file used to spawn the transmitter in the Gazebo world. After loaded The ARVA transmitter in a Gazebo scene, it shares its pose using the `tf` ROS package [18], commonly used to keep track of the relationship between multiple coordinate frames over time. In this way, the transmitter retrieves its pose in the Gazebo environment and updates a dynamic transform between the specified `frame_id` and the transmitter frame. As consequence, the `/tf` topic is automatically published by the ROS transform toolbox carrying out a message of type: `tf2_msgs::TFMessage`[4]. This message is composed by a variable length array of `geometry_msgs::TransformStamped` messages describing the pose of the transmitter and a reference to the emitting device.

The ARVA system is completed with the receiver plugin. This plugin is characterized by the following parameters. The `<channels>` specifies the number of transmitters detected by the receiver at same time, the `<parent_frame>`, that is the point of reference for output ARVA data and the `<sensor_name>` representing the device name. The

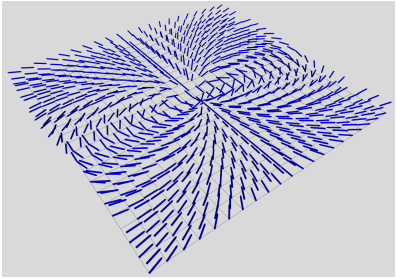

Fig. 5: Generated magnetic field displayed in RViz.

[3]http://wiki.ros.org/xacro

[4]http://docs.ros.org/jade/api/tf2_msgs/html/msg/TFMessage.html

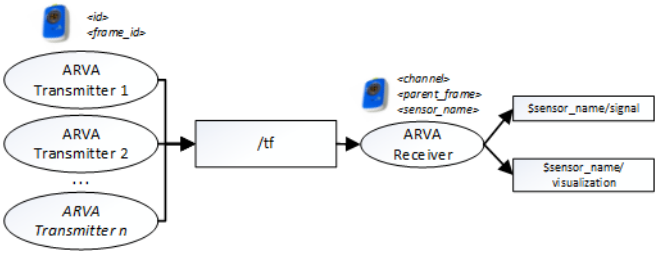

Fig. 6: arva_sim ROS package software architecture.

`sensor_name` parameter influences the name of the topic where the output of the sensor is published. This is useful when multiple ARVA receivers are spawn in the same simulation scene.

As for the output of the ARVA system, two ROS topics have been deployed to publish its output. First of all, the `/$sensor_name/signal` topic contains a variable legnth set of `arva_sim::arva_data` message. The size of this set is equal to the number of channels of the receiver. As shown in the architecture depicted in Figure 6, the receiver module subscribes to the `/tf` topic and calculates the magnetic field flux for all the transmitters spawned in the simulation scene. Considering the ARVA model described in Section II, only the closest transmitters to the given receiver are returned as output. For this reason, the detected transmitter signals are sorted with respect to their distance from the receiver and only the first -`$channels`- signals are included in the output topic. In the following, an output example of the ARVA system is reported. In particular, an instance of a message published on the `/receiver1/signal` topic after spawned one transmitter and one receiver in the simulation scene.

```
header:
  seq: 62
  stamp:
  secs: 95
  nsecs: 404000000
  frame_id: "world"
  arva_signals:
    -
    id: 2
    distance: 1046.0
    direction:
      x: 9.99855692498e-05
      y: -6.97877478029e-10
      z: 6.61148733343e-05
```

This message contains all the information needed to follow the electromagnetic flux lines emitted by the transmitters. In fact, the `id` field identifies the transmitter device. In case of invalid (or non-existent) transmitter signal, it is set to -1, while the `distance` field contains the distance between the emitter and receiver in centimeters. Regarding the framerate, the ARVA receiver provides a feedback every second (1 Hz). Finally, by following the 3D vector of the *direction* field, the receiver is able to reach the transmitter. Moreover, an additional ROS topic has been set to display the output of the receiver sensor. In particular, the `/$sensor_name/visualization` topic contains

a `visualization_msgs::MarkerArray`[5] and allows the visualization of ARVA data on RViz which are represented as arrows. The content of this topic has already been show in Figure 4 where the ARVA receiver was configured to detect four different transmitters simultaneously.

IV. Case Study

This section presents an application for autonomous robots, and in particular for quad–copters, exploiting the ARVA system in S&R operations in avalanches. The design of search strategies for improving the S&R operation in avalanche scenarios is still an open topic and the comparison of the obtained performance must be made by means of reliable tools. So, the goal of this section is to provide the implementation of a search approach, inspired by the technique adopted by human rescuers to test the reliability of the proposed ROS package `arva_sim`. Once that the first valid ARVA signal is detected, the rescuers start following the electromagnetic flux lines emitted by the ARVA transmitter. Since the flux lines pass through the transmitter, the rescuers use to move along them to reach the source of the electromagnetic field. In this context, the angle δ is used to navigate the electromagnetic field, whereas he distance d is used to regulate the navigation speed. In particular, the rescuers slowdown while getting closer to the transmitter to make the identification of the transmitter location more accurate.

This case study proposes an automatic control strategy driving the quad-copter along the ARVA flux lines. We assume that the robot is equipped with a low level control system able to follow an inertial trajectory and actuated by the inertial speed ${}^{i}V_i \in \mathbb{R}^3$, described in the inertial frame $\mathcal{F}_i$. In addition, the robotic agent is also equipped with a sensor providing the height above the soil. Given the terrain altitude, namely s, described in the inertial frame $\mathcal{F}_i$ by the map $S : \mathbb{R}^2 \mapsto \mathbb{R}_{>0}$, with $s = S(L_{xy}\,{}^{i}p_r)$, the constraint dynamics of the robotic agent is described by the system of Eq. (9):

$$\begin{aligned} {}^{i}\dot{p}_r &= {}^{i}V_i \\ {}^{r}h &= {}^{r}R_i\,{}^{i}R_t\,{}^{t}h\left({}^{t}R_i({}^{i}p_r - {}^{i}p_t)\right) + Q\,{}^{r}R_i\,{}^{i}w \\ y_s &= L_z\,{}^{i}p_r - S(L_{xy}\,{}^{i}p_r) + \nu \\ L_z\,{}^{i}p_r &\geq S(L_{xy}\,{}^{i}p_r) \end{aligned} \tag{9}$$

where, y_s models a device belonging to the family of the range sensors, which are typically corrupted by a white noise, namely ν. The high level control system proposed in this paper is composed by a *height–hold* controller to the vertical position, and one steering controller to drive the drone along the ARVA flux lines. Given the measurement ${}^{r}h$, this controller determines the first two components of the speed ${}^{i}V_i$ as:

$${}^{i}V_{xy} = L_{xy} v_{\max} \frac{k_c \|{}^{r}h\|^{-\frac{1}{3}}}{\sqrt{1 + (k_c \|{}^{r}h\|^{-\frac{1}{3}})^2}}\,{}^{i}R_r \frac{{}^{r}h}{\|{}^{r}h\|} \tag{10}$$

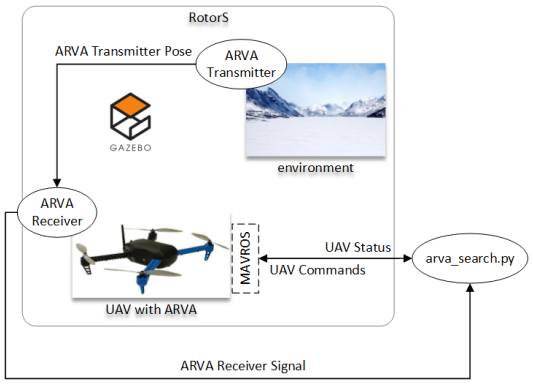

Fig. 7: Software architecture deployed for the testing of the ARVA system.

where $v_{\max} > 0$ and $k_c > 0$. The term $v_{\max} \frac{k_c \|{}^{r}h\|^{-\frac{1}{3}}}{\sqrt{1 + (k_c \|{}^{r}h\|^{-\frac{1}{3}})^2}} \in [0, v_{\max}]$ indicates the speed magnitude whereas the term ${}^{i}R_r \frac{{}^{r}h}{\|{}^{r}h\|}$ determines the orientation, in the inertial frame, of the speed vector. Given the measurement y_s and a constant reference height $r_s > 0$ the controller for the management of the height provides the vertical speed, ${}^{i}V_z$, given by the following relation ${}^{i}V_z = k_z(r_s - y_s)$. The composition of the considered laws of equations provides the overall control system, thus represented by: ${}^{i}V_i = \begin{bmatrix} {}^{i}V_{xy} \\ {}^{i}V_z \end{bmatrix}$.

This control system has been deployed to demonstrate the effectiveness of the sensor model. The proposed test case consists in a quad–copter equipped with an ARVA receiver operating in a mountain covered in snow. To simulate the quad–copter, We rely on RotorS [19], a Micro Aerial Vehicle (MAV) simulation framework providing UAV models and controllers simulated in Gazebo. In particular, the description model of the UAV provided with RotorS has been modified to carry an ARVA receiver. Additional information about how to install and start RotorS with an ARVA based robot can be found at the wiki page of `arva_sim` package: https://github.com/jocacace/arva_sim/wiki/ARVA-Gazebo-ROS-plugins.

Regarding the ARVA system, in order to use receiver data to control the motion of the robot, one or more transmitters must be spawned in the Gazebo world. To do this, the launch files located in the `arva_sim` package can be used. The software architecture deployed to evaluate the system is depicted in Figure 7 and it is discussed in the remaining of this section. Gazebo provides the information needed to initialize the sensors of the UAV, as its Inertial Measurement Unit (IMU) and GPS while the mavros[6] package retrieves and shares UAV information using ROS network enabling the control of the robot via services and topics.

The operation needed to initialize the UAV and perform the search pattern are managed by a `python` script (`arva_search.py`, located in the *script* folder of the

[5] http://wiki.ros.org/visualization_msgs

[6] http://wiki.ros.org/mavros

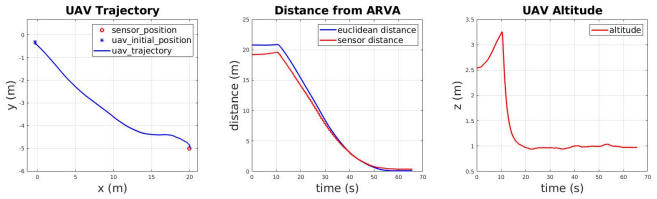

Fig. 8: I test case. ARVA position: (20.0, 5.0, 0.0)

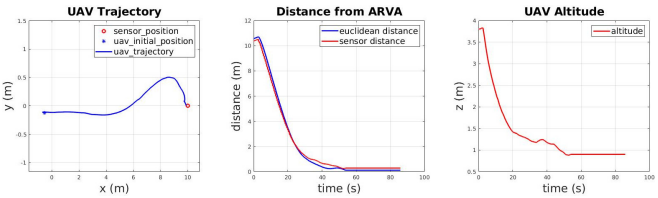

Fig. 9: II test case. ARVA position: (10.0, 0.0, 1.0)

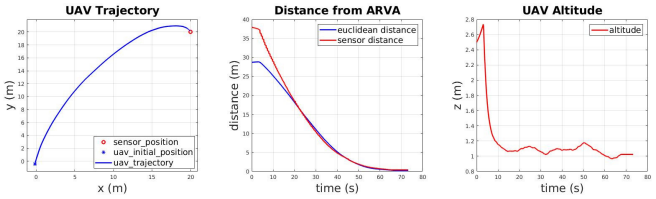

Fig. 10: III test case. ARVA position: (20.0, −20.0, −1.0)

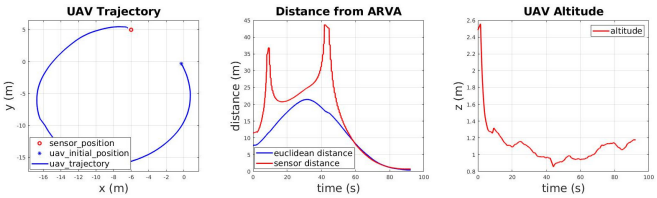

Fig. 11: IV test case. ARVA position: (−6.0, 5.0, 0.0)

`arva_sim` package). In this context, after terking off, the UAV waits for a valid ARVA signal. When a valid signal has been engaded, the controller makes the drone following the direction of the electromagnetic flux line considering the equations previously described. As for the maximum velocity of the UAV, it has been set to 1 m/s. This velocity is compatible with the one of the human rescuers. As for the ARVA sensor evaluation, We considered four different location of the workspace for the ARVA transmitters. For each test case, only one transmitter per time is spawned in the simulation. In each test, the initial position of the UAV has been set to the origin, $x = 0$ and $y = 0$, whereas for the altitude, the initial takeoff height of the quad–copter has been randomly set up to 4 meters. Regarding the height reference, r_s, the presented results have been obtained by imposing a fixed altitude of $r_s = 1$ meter. In this way, the quad–copter is able to safely navigate the environment without crashing on the terrain.

During each test the 2D trajectory of the UAV and the distance from the receiver sensor were saved. The proposed test case demonstrates the working principle of the ARVA plugins. The data monitored during the four scenarios are reported in Figures 8-11. From such graphics it is possible to see that the UAV is able to reach the transmitter following the electromagnetic field flux lines. In particular, the shape of the trajectories performed by the UAV recall the magnetic field shape. Regarding the figures depicting the distance from the ARVA transmitter, can be noticed that following the electromagnetic field lines does not implies that the transmitter is reached with the shortest path. Indeed, for all the experiments the Euclidean distances show a limited growth before decreasing to small values close to zero. This is particularly evident in the fourth test (see 11).

V. Conclusions

This paper proposed a new ROS package called `arva_sim` (`https://github.com/jocacace/arva_sim`) who aims to provide the simulation, the configuration files and libraries to emulate a complete ARVA system. The ARVA represents the forefront technology adopted in S&R operations to localize victims of avalanches buried under the snow. This system is composed by two modules: the transmitter and receiver. The transmitter generates a magnetic field detected by the receiver. The simulation package is based on ROS Gazebo plugins. In this way, developers and researchers can equip simulated robots with ARVA sensors and designing faster and smarter S&R strategies. Finally, a use case in which an autonomous navigation mission based on ARVA data has been presented in order to validate the accuracy and effectiveness of the package `arva_sim`.

APPENDIX

This appendix contains a brief tutorial to download, compile and run an initial example to test the working principle of the proposed system. We assume that a Linux Operating system with an installation of ROS *Kinetic* or *Melodic* is used. To download and install `arva_sim` move in your ROS package workspace and execute the following commands:

```
$ git clone https://github.com/jocacace/arva_sim
```

Select the desired branch: `arva_sim` repository has two main branches. The default branch supports Gazebo 9 (and consequently ROS *Melodic*). To use `arva_sim` with ROS *Kinetic* type the following commands:

```
$ roscd arva_sim
$ git checkout gazebo7-devel
```

Finally, compile the package:

```
$ cd ~/catkin_ws
$ catkin_make
```

After compiled the package, use the following command to start the whole system. Note that some of the considered commands could lock the terminal command. For this reason, could be useful to use multiple terminals.

```
$ roslaunch gazebo_ros empty_world.launch
$ roslaunch arva_sim spawn_arva_transmitters.launch
$ roslaunch arva_sim spawn_receiver.launch
```

Other than load the ARVA receiver and four transmitters in the Gazebo environment, the last command opens the RViz program in order to show its output using a set of visual markers as reported in Figure 4.

References

[1] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. Hoepflinger, "Anymal - a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 38–44, Oct 2016.

[2] N. Kashiri, L. Baccelliere, L. Muratore, A. Laurenzi, Z. Ren, E. M. Hoffman, M. Kamedula, G. F. Rigano, J. Malzahn, S. Cordasco, P. Guria, A. Margan, and N. G. Tsagarakis, "Centauro: A hybrid locomotion and high power resilient manipulation platform," *IEEE Robotics and Automation Letters*, vol. 4, pp. 1595–1602, April 2019.

[3] M. Konyo, Y. Ambe, H. Nagano, Y. Yamauchi, S. Tadokoro, Y. Bando, K. Itoyama, H. Okuno, T. Okatani, K. Shimizu, and E. Ito, *ImPACT-TRC Thin Serpentine Robot Platform for Urban Search and Rescue: Results from the ImPACT Tough Robotics Challenge*, pp. 25–76. 01 2019.

[4] C. Sampedro, A. Rodriguez-Ramos, H. Bavle, A. Carrio, P. de la Puente, and P. Campoy, "A fully-autonomous aerial robot for search and rescue applications in indoor environments using learning-based techniques," *Journal of Intelligent & Robotic Systems*, vol. 95, pp. 601–627, Aug 2019.

[5] D. Falanga, K. Kleber, S. Mintchev, D. Floreano, and D. Scaramuzza, "The foldable drone: A morphing quadrotor that can squeeze and fly," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, 12 2018.

[6] L. Marconi, C. Melchiorri, M. Beetz, D. Pangercic, R. Siegwart, S. Leutenegger, R. Carloni, S. Stramigioli, H. Bruyninckx, P. Doherty, A. Kleiner, V. Lippiello, A. Finzi, B. Siciliano, A. Sala, and N. Tomatis, "The sherpa project: Smart collaboration between humans and ground-aerial robots for improving rescuing activities in alpine environments," in *2012 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp. 1–4, Nov 2012.

[7] V. Ferrara, "Technical survey about available technologies for detecting buried people under rubble or avalanches," pp. 91–101, 05 2015.

[8] L. Joseph and J. Cacace, *Mastering ROS for Robotics Programming - Second Edition: Design, Build, and Simulate Complex Robots Using the Robot Operating System*. Packt Publishing, 2nd ed., 2018.

[9] S. Waharte and N. Trigoni, "Supporting search and rescue operations with uavs," 09 2010.

[10] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. L. Grixa, F. Ruess, M. Suppa, and D. Burschka, "Toward a fully autonomous uav: Research platform for indoor and outdoor urban search and rescue," *IEEE Robotics Automation Magazine*, vol. 19, pp. 46–56, Sep. 2012.

[11] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L. Cooper, M. Quigley, J. A. Adams, and C. Humphrey, "Supporting wilderness search and rescue using a camera-equipped mini uav: Research articles," *J. Field Robot.*, vol. 25, pp. 89–110, Jan. 2008.

[12] J. Cacace, A. Finzi, V. Lippiello, M. Furci, N. Mimmo, and L. Marconi, "A control architecture for multiple drones operated via multimodal interaction in search amp; rescue mission," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp. 233–239, Oct 2016.

[13] J. Cacace, A. Finzi, and V. Lippiello, "Implicit robot selection for human multi-robot interaction in search and rescue missions," in *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 803–808, Aug 2016.

[14] G. Bevacqua, J. Cacace, A. Finzi, and V. Lippiello, "Mixed-initiative planning and execution for multiple drones in search and rescue missions," vol. 2015, 07 2015.

[15] P. Pinies and J. D. Tardos, "Fast localization of avalanche victims using sum of gaussians," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 3989–3994, May 2006.

[16] H. Kam, S.-H. Lee, T. Park, and C.-H. Kim, "Rviz: a toolkit for real domain data visualization," *Telecommunication Systems*, vol. 60, pp. 1–9, 10 2015.

[17] L. Kunze, T. Roehm, and M. Beetz, "Towards semantic robot description languages," in *2011 IEEE International Conference on Robotics and Automation*, pp. 5589–5595, May 2011.

[18] T. Foote, "tf: The transform library," in *2013 IEEE Conference on Technologies for Practical Robot Applications (TePRA)*, pp. 1–6, April 2013.

[19] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, *RotorS âĂŞ A Modular Gazebo MAV Simulator Framework*, vol. 625, pp. 595–625. 01 2016.