Search and Rescue of Avalanche Victims

1st Dongyue Lu
Department of Informatics
Technical University of Munich
Munich, Germany
dongyue.lu@tum.de

2nd Yamo Akrami
Department of Informatics
Technical University of Munich
Munich, Germany
ga63hen@mytum.de

3rd Xuhui Zhang
Department of Mechanical Engineering
Technical University of Munich
Munich, Germany
xuhui.zhang@mytum.de

4th Yunfeng Kang
Department of Informatics
Technical University of Munich
Munich, Germany
yunfeng.kang@tum.de

5th Yuhang Cai
Department of Informatics
Technical University of Munich
Munich, Germany
yuhang.cai@tum.de

***Abstract*—This work explores the possibility of using UAVs in avalanche rescue. We deploy a UAV equipped with an avalanche beacon in a simulation environment. Four different algorithms are proposed for the search of victims in avalanche scenario, and methods' performances are compared through a series of experiments. The conclusion demonstrates the high efficiency, high accuracy and high robustness of victim rescue using drones, which has important implications for the practical application of UAVs.**

***Index Terms*—Avalanche Rescue, Triangulation, Iterative local search, Drone path planning**

I. Introduction

Sport activities such as skiing, snowboarding or hiking in winter are very popular [1] [2]. A potential danger that winter sportsmen are confronted with is avalanches, and they pose a great threat since they happen most of the time in unguarded, unwatched and less accessible areas and the process of locating avalanche victims takes a long time. Moreover, the victim survivability drops below 80% after only 10 min of being buried according to an avalanche survivability survey [3]. Hence, it is very important to locate avalanche victims fast and accurately.

An area hit by an avalanche is difficult to access and poses sometimes additional danger to the rescue team. But drones integrated with automatic devices are fast and agile which makes them a perfect rescue agent in avalanche scenarios. They are often used to locate the victim stuck in the avalanche autonomously and quickly. Many previous literature and projects [4] [5] [6] have demonstrated the feasibility of using drones in rescues.

In this work we address the rescue problem of victims in an avalanche scenario. We mainly focus on how to locate victims quickly and accurately. Further, we conduct our work in a simulation environment and deploy a drone with a sensor to locate avalanche victims. In Section II, we describe our simulated avalanche scenario and the problem addressed in this work. Following, in Section III we discuss the specification of the sensor model which receives the emitted signals by avalanche victims. Search methods to cover the avalanche area and locate the victims are addressed in Section IV. In section V we introduce our trajectory generation method and controller. The experiments and results are presented in VI. Finally, in Section VII we will summarize the work and give an outlook for perspective research.

II. Problem Description

The avalanche scene is simplified to a slope with length and width of several hundred meters with an angle $\theta = 7°$. Victims are distributed on the slope and the drone flies at a fixed height $h = 5$m over the slope. The drone is a quadrotor unmanned aerial vehicle (UAV) and its dynamics are similar to that formulated by Taeyoung Lee et al. in [7]. It carries a simulated avalanche beacon sensor to receive the victim's signal. It should cover the entire avalanche area with least time, accurately locate the victim, and fly to the victim afterwards to show that the localization is successful. We stipulate that the rescue must be within 15 minutes to improve the survival probability of the victim. For simplicity, we do not consider external environmental factors such as temperature, altitude, and wind, nor do we consider factors such as drone battery capacity and weight.

III. Sensor Model

Avalanche transceivers are common devices employed in mountain activities. In transmission mode, they emit a low pulsed radio signal at frequency $f = 457Hz$. Once commutated into the receiving mode, they work as radio direction finding equipments reconstructing the direction and the distance of the source transceiver, to search signals coming from other transceivers.

In this work, we assume that the sensor can only give the signal's intensity which is inversely proportional to the distance. Except the method of IV-B3, we assume that the sensor can only receive the intensity signal. For IV-B3, the sensor can also receive the direction signal. The transmission range of different transceivers varies from $20m$ to $70m$. and in this work, we set the working range $r = 20m$, so the observed intensity of the signal emitted by a victim is computed by

$$\bar{i}_t = \frac{r}{dist_v} \tag{1}$$

where $dist_v$ is the ground truth distance to a victim.

A. Noise

We assume that the intensity at timestamp t is noisy and defined as follows

$$i_t = \bar{i}_t + \epsilon_t \tag{2}$$

where ϵ_t is the noise term that follows an exponential distribution scaled by uniform distribution

$$\epsilon_t \sim U([0,1]) \cdot \lambda e^{-\lambda \bar{i}_t} \tag{3}$$

Exponential distribution of the noise comes from the behaviour of sensors in the real world. If the drone flies away from a victim, the intensity decreases and the noise increases. In contrast, approaching a victim makes the intensity increase and the noise decrease. Moreover, the multiplication with an uniform distribution $U([0,1])$ in Equation (3) introduces a randomness in the noise. Furthermore, we set $\lambda = 1$.

B. Direction Finding

In IV-B3 we introduce an approach that relies on the magnitude of the intensity in each direction. Such a sensor model can be interpreted as an antenna that points in positive and negative x, y and z direction (see Figure 6 in Appendix B). In each direction the sensor measures a intensity between $[-1, 1]$.

We reach this by computing

$$\hat{d}_t = \frac{\vec{d}_t}{\|\vec{d}_t\| + c} \tag{4}$$

with c as a constant and $\vec{d}_t = \vec{v}_t - \vec{x}_t$. Here, x_t is the current position of the drone and v_t is the victim position.

IV. Search Methods

Due to the limited working range of the transceiver, we have to get close enough to the victim for precise positioning. So our search is divided into global search and local search.

A. Global Search

For global search, to cover the search area as quickly as possible, we use the most common method used by rescue teams: in a simplified scenario as Figure 1, the drone flies in the normal direction of the avalanche, and after reaching the end point, the drone turns 90°, after flying a distance of grid length w_g in the parallel direction of the avalanche, turns 90° again and flies in the normal direction of the avalanche. The grid length w_g depends on specific local search method. The drone should repeat this procedure to cover the entire avalanche area.

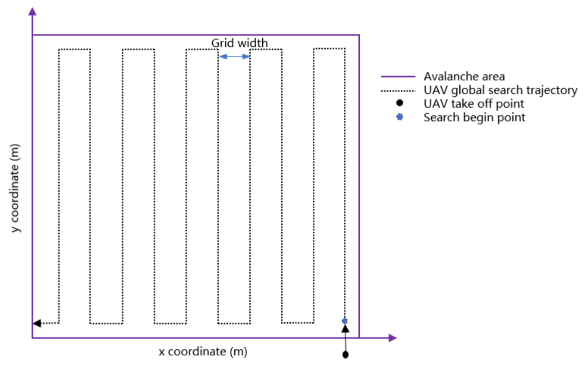

Fig. 1. Global Search

B. Local Search

We try four different local search methods separately.

1) Cross Flight: The basic idea of Cross Flight method comes from [4], we expand this method to multiple victims and 3D scenario. The whole process is shown in Figure 2, Once the victim signal is received, the drone enters the Cross Flight local search, records the position $h1$ at this time, and flies forward until the victim signal is lost, and the disappearance position is recorded as $h2$. Then the drone flies backwards to the midpoint between $h1$ and $h2$, turns 90°, repeats the above process, records the disappearance positions of the two victim signals in the vertical direction as $v1$ and $v2$, and records the midpoint of $v1$ and $v2$ as p_c. p_c must be the projected point of victim p_v on the drone plane. $h1$, $h2$, $v1$, $v2$ must be on the sphere with the victim position p_v as the center, and these four points are on the circle with p_c as the center of the circle and $\vec{\mathbf{n}}_\mathbf{c}$ as the normal vector. Since the victim plane is known, based on a certain point p_p on the plane and its normal vector $\vec{\mathbf{n}}_\mathbf{p}$, the intersection of the line $\vec{\mathbf{n}}_\mathbf{c}$ and the victim plane can be calculated as the victim's position.

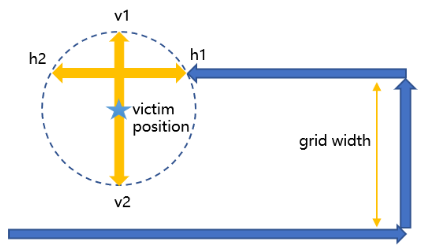

Fig. 2. Cross Flight

$$t = \frac{\vec{\mathbf{n}}_\mathbf{p}(p_p - p_c)}{\vec{\mathbf{n}}_\mathbf{p} \cdot \vec{\mathbf{n}}_\mathbf{c}} \tag{5}$$

$$p_v = p_c + \vec{\mathbf{n}}_\mathbf{c} t \tag{6}$$

It can also be seen from the Figure 2 that for the Cross Flight method, in the extreme case, $w_g = 2r = 40m$, in practice, we choose $w_g = 35m$ to cover a larger search area as much as possible, and reduce the possibility of missing victim signal.

2) Iterative Cross Flight (ICF): After the local search, the distance error could be not low enough due to the heavy noise. Therefore, it is necessary to further take a step to gradually reduce the error to a certain level through an iterative method. Here we use the intensity received at the victim position as the criterion to decide whether to let the drone enter Iterative Cross Flight.

As shown in the Figure 3, there are two unit vectors $\overrightarrow{e1}$ and $\overrightarrow{e2}$ on the drone plane, which are the movement directions of the drone during Cross Flight and Iterative Cross Flight. After the Cross Flight, if the intensity received at the victim location is not higher than the threshold, the drone will fly from the victim position to the midpoint p_c in the drone flight plane. Next, instead of continuing the global search, it will repeat the Cross Flight in this plane. In the Figure 3, $v1$ and $v2$ are the boundary points in Cross Flight and $v3$ and $v4$ are the boundary points in the first iterative search. Unlike the normal Cross Flight, the criterion for judging whether the drone should stop is no longer an intensity of zero, but an appropriate threshold. This threshold increases with the number of iterations. This allows the drone to perform local search in a smaller area, thereby improving the accuracy of the results.

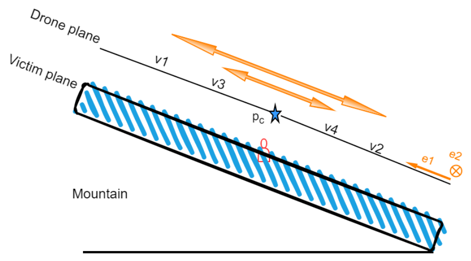

Fig. 3. Iterative Cross Flight

3) Iterative Local Search (ILS): The idea of the Iterative Local Search algorithm is to navigate the drone gradually to the victim. Preferably, the drone should take large steps towards a victim when far away and small steps, otherwise. Hence, we exploit the intensity of the sensor model described in III to influence the step size in each iteration.

One solution is to exploit the direction and the magnitude of the intensity observed by the sensor model. In (7) we see an iterative local search method with an adaptive step size assuming a sensor model discussed in III-B.

$$\vec{x}_{t+1} = \vec{x}_t + \alpha * \frac{r_t}{i_t} * \hat{d}_t \tag{7}$$

Here, $\vec{x}_{t+1}$ is the next state at timestamp $t+1$ and $\vec{x}_t$ is the current state at timestamp t and $\hat{d}_t$ is computed using Equation (4), $\frac{r_t}{i_t}$ is the adaptive step size, and α is a scaling factor. We found out that $\alpha = 0.5$ prevents the drone from overshooting or oscillating. The step size depends on the range r_t and the intensity i_t. It decreases if the drone approaches a victim because the intensity increases. It also prevents the drone to take big steps as this could lead to overshooting or oscillating behaviour. On the other hand, if the drone is far away from a victim the step size is large because the intensity is low. Hence, the drone is taking big steps towards a victim and preventing a slow convergence. The algorithm terminates when $\|\vec{x}_{t+1} - \vec{x}_t\|_2 \leq 0.1$.

For an attempt to devise an approach without taking advantage of the direction see in Appendix A..

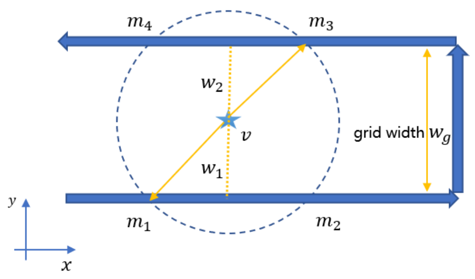

Fig. 4. Triangulation

4) Triangulation: Different from the above three methods, triangulation only records the position where the drone receive or lose the victim's signal during the global search process, which are denoted as marginal points m_i. After covering all area, the drone calculates the position of all victims and returns to rescue. Since the marginal points of all victims are mixed together, we need to find out the correspondence between each victim and its marginal points. We perform two screenings, as shown in Figure 4. First, the midpoint of a victim's marginal points in the x-coordinate must be the same as the victim's x-coordinate

$$v_x = \frac{1}{2}(m_{1x} + m_{2x}) = \frac{1}{2}(m_{3x} + m_{4x}) \tag{8}$$

After screening candidates, given the grid width w_g, we conduct a second screening: The candidates must be on the victim-centred circle:

$$(v_x - m_{1x})^2 + {w_1}^2 = (v_x - m_{3x})^2 + {w_2}^2 \tag{9}$$

$$w_1 + w_2 = w_g \tag{10}$$

Through the above two screenings, we can determine all pairs of marginal points belonging to the same victim and calculate the victim's position, but it should be noted that this point is actually the projection point of the victim's position on the drone plane. In the same way as in the previous methods, given drone plane and victim plane, we can calculate the real victim position from this projection point.

After locating the victims, the drone visits corresponding points and records the signal intensity at each point. This step works as verification and rescuing. If it is larger than a given threshold, we assume the victim is actually located here and is rescued. To reduce the influence of the signal's noise, we can set the grid width w_g as smaller values, which leads to more measurements for each victim and more accurate results.

V. Path Planner and Controller

A. Potential Field

In this work, we try to use potential field to navigate the drone to the victim after locating the victim. Regardless of obstacles, we only need to build an attraction field centered on the target location

$$U_t = \frac{1}{2}\zeta(\vec{x}_{goal} - \vec{x}_t)^2 \tag{11}$$

The position update depends on the attractive force, which is the derivative of the potential energy field.

$$\vec{x}_{t+1} = \vec{x}_t + \nabla U_t \tag{12}$$

$$\nabla U_t = \zeta(\vec{x}_{goal} - \vec{x}_t) \tag{13}$$

The planner terminates when $\|\vec{x}_{t+1} - \vec{x}_t\|_2 \leq 0.5$.

B. Trajectory Generation

Since we don't need to consider obstacles, all trajectories can run in a straight line, so we use a simple trajectory planner to fly to the target point in a straight line at a given speed. Based on the velocity and kinematic equations, we can calculate the desired state of the drone for each step and pass it to the controller. When approaching the target, we use recursive judgment of the distance, and when the distance from the target point increases, we replan the trajectory to prevent overshoot.

C. Geometric Tracking Controller

We use a geometric tracking controller [7] that is designed so that the position tracking error converges to zero when there is no attitude tracking error, and it is properly adjusted for non-zero attitude tracking errors to achieve asymptotic stability of the complete dynamics. This controller can achieve almost-global convergence for the drone.

TABLE I
Comparsion between different local search methods

method	without direction			with direction
	Cross Flight	ICF	Triangulation	Dir. ILS
Error (m)	0.33	**0.22**	0.47	0.36
Area (m^2)	415*300	415*300	415*300	415*300
Time (s)	**733**	796	864	**702**
Effi. (m^2/s)	**186**	172	158.5	**195**
HA	94	**99**	92	**100**
LA	5	**0**	8	**0**
Failure	1	1	**0**	**0**

VI. Experiments and Results

We compare the efficiency, accuracy and robustness of the different search methods described above. In order to ensure the fastest rescue speed, we stipulate that the speed of global search v_g is $8m/s$. After locating the victim, the speed of flying to the victim v_l is $5m/s$. We do not hover near the victim to reduce time waste. We use the victim coverage area divided by the rescue time to represent the efficiency metrics of rescue (denoted as Effi. in the table). For accuracy and robustness, we define that there are ten victims in each rescue, and classify the rescue performance with different errors: the distance error less than $1m$ is highly accurate, the distance error between $1-5m$ is low accurate (denoted as HA and LA in the table), and higher than $5m$ means the search fails. We perform ten rescues per method using randomly generated victim locations, i.e. a total of 100 victims, and record the average distance error, total rescue time, covered area, and the number of different cases. The experiment results are shown in Table I.

From the comparison we can see that between methods without direction, ICF further iterates on the already high accuracy of Cross Flight and thus achieves the highest accuracy. However, these two methods are less robust, which is related to the fact that these two methods need to fly repeatedly to accept the victim's noisy signal. Especially when the victim is at the edge of the transceiver's working range, weak signal and repeated flight are likely to cause the method to fail. Triangulation consumes the most time due to the need for shorter grid width and increased flight distance. Further comparison between Cross Flight and triangulation can be seen in Appendix C.

We list the directed ILS separately, because it has direction as more conditions. We can see that with direction, after receiving the signal, this method can fly directly to the victim, saving time and reducing the possibility of interference by noise. The design of adaptive size also reduces the occurrence of overshoot and oscillation, and has strong robustness.

Compared with human, the flight speed of a drone at 8m/s is about 4 times that of the human walking speed, and the search for an area of about 120,000 square meters can be completed in about 12 minutes, especially after having the direction information, the drone can directly fly to the victim, which greatly saves time and effectively increases the victim's chance of survival.

VII. Conclusions

UAV has great application potential in avalanche rescue. In this project, we use different methods to achieve accurate and rapid positioning and rescue of victims in avalanche scenes. We compare the efficiency, accuracy and robustness of different methods. The results show that iterative methods with more sensor information have higher efficiency and robustness. Under the condition of less sensor information, the traditional geometric method has higher accuracy. The search method can further improved by exploring field-based direct search methods. At the same time, the coverage efficiency of UAVs has been proved to be much higher than that of humans, which provides a simulation basis for future practical applications.

References

[1] Kurt Winkler, Adrian Fischer, and Frank Techel. Avalanche risk in winter backcountry touring: Status and recent trends in switzerland. *International Snow Science Workshop 2016 Proceedings, Breckenridge, CO, USA*, pages 270–276, October 2016.

[2] Stephan Harvey and Benjamin Zweifel. New trends of avalanche accidents in backcountry terrain in switzerland. *Proceedings Whistler 2008 International Snow Science Workshop September 21-27, 2008*, page 900, September 2008.
[3] Pascal Haegeli, Markus Falk, Hermann Brugger, Hans-Jrg Etter, and Jeff Boyd. Comparison of avalanche survival patterns in canada and switzerland. *CMAJ : Canadian Medical Association journal = journal de l'Association medicale canadienne*, 183:789–95, 03 2011.
[4] Mario Silvagni, Andrea Tonoli, Enrico Zenerino, and Marcello Chiaberge. Multipurpose uav for search and rescue operations in mountain avalanche events. *Geomatics, Natural Hazards and Risk*, 8(1):18–33, 2017.
[5] Pietro Iob, Luca Frau, Piero Danieli, Lorenzo Olivieri, and Carlo Bettanini. Avalanche rescue with autonomous drones. In *2020 IEEE 7th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, pages 319–324, 2020.
[6] Ilario Antonio Azzollini, Nicola Mimmo, Lorenzo Gentilini, and Lorenzo Marconi. Uav-based search and rescue in avalanches using ARVA: an extremum seeking approach. *CoRR*, abs/2106.14514, 2021.
[7] Taeyoung Lee, Melvin Leok, and N. Harris McClamroch. Geometric tracking control of a quadrotor uav on se(3). In *49th IEEE Conference on Decision and Control (CDC)*, pages 5420–5425, 2010.

Appendix A
Undirected ILS

In Equation (14) we see our first attempt to devise a state update equation.

$$\vec{x}_{t+1} = \vec{x}_t + I * \Delta\vec{x} \tag{14}$$

$I = \tanh(\Delta i)$ where $\Delta i = i_t - i_{t-1}$ is the difference of previous and current intensity. This term influence the forward and backward movement. Further, $\Delta\vec{x} = \vec{x}_t - \vec{x}_{t-1}$ is the difference of the previous and current state of the drone and acts as an step size.

The idea behind this update rule is that the drone will keep flying in the direction when the argument of the tangens hyperbolicus is greater than zero. This happens when the difference of the intensity Δi is greater than zero. If the difference of the intensity Δi is smaller than zero, the drone will fly in the opposite direction (see Figure 5). Lastly, we multiply the difference of the state of the drone $\Delta\vec{x}$ in order to prevent large steps which leads to overshooting or oscillating. As in IV-B3 the algorithm terminates when $\|\vec{x}_{t+1} - \vec{x}_t\| \leq 0.1$.

However this approach has a downside. Imagine the drone is flying along the x-axis so that there is no change in y-direction of the state. This leads to $\Delta y = 0$ and y_t will stay the same for every iteration in the update rule (14).

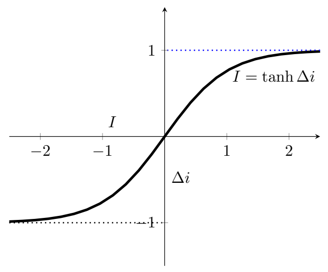

Fig. 5. The difference of the current and previous intensity passed to tangens hyperbolicus function.

Appendix B
Figures

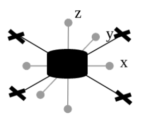

Fig. 6. An antenna with 6 end points directed in positive x, y, z and negative x, y and z.

Appendix C
Cross Flight and Triangulation

Cross Flight, ICF and Dir. ILS interrupt global search and enter local search when receiving victim signal. However, in triangulation victims are localized after the drone cover the whole area. Results of Cross Flight and triangulation tell advantages of each method. For large area with few victims, Cross Flight achieves less time. This comes from larger grid width.

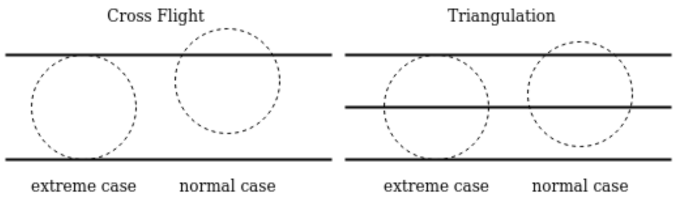

Fig. 7. To receive signal, grid width for Cross Flight should be no larger than the diameter. However, triangulation needs at least 4 marginal points for localization. The maximal grid width is the radius. Cross Flight needs about half fly-overs of triangulation.

Whereas, triangulation is more competitive as victims increase in the same area due to a different rescuing strategy. Local search time increases obviously for Cross Flight, while total time of triangulation only increases slightly. This comes from more calculation time and more victims to finally go through.

TABLE II
Comparison between Cross Flight and triangulation

	10 victims	15 victims	20 victims	25 victims
Time of Cross Flight [s]	194	307	409	521
Time of triangulation [s]	277	326	380	445

The test area is 144*300 m^2. From 20 victims triangulation overtakes Cross Flight with less time.