EPFL

Thèse n° 7606

Source Term Estimation Algorithms for Gas Sensing Mobile Robots

Présentée le 17 septembre 2021

Faculté de l'environnement naturel, architectural et construit
Laboratoire de systèmes et algorithmes intelligents distribués
Programme doctoral en robotique, contrôle et systèmes intelligents

pour l'obtention du grade de Docteur ès Sciences

par

Faezeh RAHBAR

Acceptée sur proposition du jury

Prof. B. Faltings, président du jury
Prof. A. Martinoli, directeur de thèse
Prof. A. Lilienthal, rapporteur
Prof. K. K. Leang, rapporteur
Prof. A. Ijspeert, rapporteur

École polytechnique fédérale de Lausanne

2021

To my mother
and her endless love and patience.

To my father
and his constant support and encouragement.

Acknowledgements

Throughout my doctoral studies at EPFL, I have received a great deal of support and assistance, without which the accomplishment of this work would not have been possible, and for which I am deeply grateful. First of all, I would like to thank my advisor, Prof. Alcherio Martinoli, who gave me the opportunity to pursue this research and provided me with constant encouragement and support over the years. His professional expertise as an advisor, and his compassion as a colleague, were reassuring and heartwarming throughout the stressful process of a challenging research, especially in pandemic time.

I would like to thank the thesis committee, Prof. Boi Faltings, Prof. Auke Ijspeert, Prof. Achim Lilienthal, and Prof. Kam K. Leang, for taking the time to review this manuscript and for providing valuable feedback.

I am thankful to all the people that contributed to the research presented in this thesis. I would like to especially thank Ali Marjovi, whose advice and support had a significant impact on the success of this thesis. He guided me through the world of research in the early stages of my doctoral studies and helped me with the first steps with his expertise in mobile chemical sensing and mapping. In the later stages of my doctoral studies, I had the opportunity to have Chiara Ercolani as close collaborator. We shared many eventful days in our experimental facility and office. I am so grateful for all the good moments and laughs that we shared as well as the deep support she provided me as colleague.

I would like to thank all my collaborators at DISAL for the warm atmosphere they created in the lab and for all the good moments that we shared. I would like to express special thanks to Anwar Quraishi, with whom I shared almost the entire PhD journey; Zeynab Talebpour and Bahar Haghighat, with whom I spent so much enjoyable time; Cyrill Baumann, Kagan Erünsal, and Lucas Wälti, with whom we went through the pandemic condition while keeping our strength; and many other senior PhD students and PostDocs who inspired me with their valuable advices and experiences, Steven Roelofsen, Alicja Wąsik, Miloš Vasić, Duarte Dias, Adrian Arfire, Iñaki Navarro, José Nuno Pereira, Maria Boberg, Alexander Bahr, Felix Schill, and Thomas Lochmatter.

Furthermore, I would like to thank Corinne Farquharson for her kind support and excellent

organization skills, on which we all rely in the lab. I would also like to acknowledge Denis Rochat and Emmanuel Droz for their invaluable help on the IT and infrastructure.

Moreover, I had the pleasure to supervise talented and hard-working students during my doctoral studies. I would like to thank Romain Emery, Thierry Dubosson, Rémi Laure, Nikita Lazarev, Pierre Kibleur, Julian Ruddick, Vincent Demotz, Danjiao Ma, Quentin Golay, Lucie Houel, Juraj Korcek, Mickaël Salamin, Axel Nilsson, Hugo Grall Lucas, and Louis-Nicolas Douce.

I am especially grateful to my closest friends for lifting my spirit and encouraging me in the most difficult times, even those who were not physically with me in Lausanne. Especial thanks go to my friend Michel Moukari for his support and helpfulness.

I would like to express my deepest gratitude to my parents who always provided me with support and warmed my heart with their kindness. They were always there for me when I needed them and never lacked confidence in me, even when I did in myself. I am also thankful to my amazing brother Mahdiyar for his endless support and my sister Shokoufeh and her husband Mohamad for their kind guidance when I needed it the most.

Finally, I am deeply grateful to my husband Alireza who has been by my side throughout difficult days and brought sweetness to my life with his love and support.

Lausanne, June 18, 2021 Faezeh Rahbar

Abstract

Localizing sources of airborne chemicals with mobile sensing systems finds applications in various crucial and perilous situations, such as safety and security investigation for detecting explosives or illegal drugs, search and rescue operations to locate survivors in the aftermath of natural hazards, or environmental monitoring in unsafe sites, following harmful leaks. Using autonomous robots in such situations would eliminate or, at least, reduce human intervention and keep them from harm. Additionally, such operations would be more cost-effective and more time-efficient. That is why, in the past 30 years, gas source localization has been an attractive research topic in robotics and related areas, where different methods have been designed and evaluated for this purpose.

However, the inherent complexity of gas dispersal phenomena, which is non-trivial to analyze and predict, especially in complex environments, is the main source of challenges in this field. Therefore, researchers tend to design and evaluate algorithms in simplistic environments before tackling more complex ones.

In this thesis, we have designed and investigated a gas source localization algorithm based on source term estimation with a probabilistic approach. After validating the performance of the method in a baseline environment using a wheeled-robot, we gradually enhanced our method by enriching it with new features in order to be adaptable to more complex scenarios. In particular, the algorithm was shown to be successful in a simplified three-dimensional setup as well as in an unknown environment where no global map and positioning system is available. Furthermore, it was deployed on a homogeneous multi-robot system, where different coordination strategies between robots were designed and studied. Finally, designing a data-driven plume model and integrating it to the main framework of the method allowed for adaptation to cluttered environments.

The method is systematically evaluated through high-fidelity simulations and in a wind tunnel emulating realistic and repeatable conditions. Lastly, the performance of our algorithm was compared with other state-of-the art methods to show its potentials and limits.

Keywords: gas source localization; source term estimation; probabilistic algorithms; information-driven motion planning; chemical sensing.

Résumé

La localisation des sources de particules chimiques dispersées dans l'air avec des systèmes de détection mobiles, a de nombreuses applications potentielles dans diverses situations cruciales et périlleuses, telles que les enquêtes de sûreté et sécurité pour la détection d'explosifs ou de drogues illégales, les opérations de recherche et sauvetage pour localiser les survivants à la suite de catastrophes naturelles, ou la surveillance environementale dans des conditions dangereuses, suite à des fuites nocives. L'utilisation de robots autonomes dans de telles situations éliminerait ou, du moins, réduirait l'intervention humaine et animale et previendrait tout impact nefaste sur des êtres vivants. De plus, ces opérations seraient plus rentables et plus rapides. C'est pour cela qu'au cours des 30 dernières années, la localisation de source de gaz a été un sujet de recherche attrayant en robotique et dans les domaines connexes, où différentes méthodes ont été conçues et évaluées à cette fin.

Cependant, la complexité inhérente aux phénomènes de dispersion de gaz, qui n'est pas évident à analyser et à prévoir, en particulier dans des environnements complexes, est la principale source de défis dans ce domaine. Par conséquent, les chercheurs ont tendance à concevoir et à évaluer des algorithmes dans des environnements simplistes avant d'aborder des plus complexes.

Dans cette thèse, nous avons conçu et étudié un algorithme de localisation de source de gaz, basé sur l'estimation des termes de source (Source Term Estimation) avec une approche probabiliste. Après avoir validé les performances de la méthode dans un environnement de référence avec un robot à roues, nous avons progressivement amélioré notre méthode en l'enrichissant de nouvelles fonctionnalités, afin d'être adaptable à des scénarios plus complexes. En particulier, l'algorithme s'est avéré efficace dans une configuration tridimensionnelle simplifiée, ainsi que dans un environnement inconnu où aucune carte globale et système de positionnement n'est disponible. De plus, il a été déployé sur un système multi-robot homogène, où différentes stratégies de coordination entre robots ont été conçues et étudiées. Enfin, la conception d'un modèle de dispersion de gas basé sur des données et son intégration au cadre principal de la méthode ont permis une adaptation à des environnements encombrés.

La méthode a été systématiquement évaluée par des simulations haute fidélité et en soufflerie, émulant des conditions réalistes et reproductibles. Enfin, les performances de notre

algorithme ont été comparées à d'autres méthodes de pointe pour montrer ses potentiels et ses limites.

Mots clés : localisation de la source de gaz ; estimation des termes de source ; algorithmes probabilistes ; planification de mouvement basée sur information ; détection chimique.

Contents

Introduction Part I

1 Gas Source Localization

In 2004, three independent groups of scientists announced, for the first time, the discovery of methane in the atmosphere of Mars. The presence of methane means either presence of life, or unusual geological activities; in any case, it is of utmost interest to find the origin of the release. However, whenever methane was detected, it was rapidly removed from the atmosphere by an unidentified process [1].

15 years later, NASA's Curiosity Mars rover measured the largest amount of methane ever found during the mission, but it drastically decreased at the following experiment. "The methane mystery continues," said Ashwin Vasavada, Curiosity's project scientist at NASA's Jet Propulsion Laboratory in Pasadena, California. "We're more motivated than ever to keep measuring and put our brains together to figure out how methane behaves in the Martian atmosphere." The gas source remains unknown and under study to this day.

1.1 Motivation

Locating the source of methane release could indicate whether Mars is active biologically or geologically, which is, in both cases, a break-through in understanding outer-space nature. Back on earth, on the other hand, localizing the source of airborne chemicals can lead to saving human lives, in search and rescue operations following natural hazards such as earthquakes. It ensures security by locating the source of dangerous substances such as explosives or drugs. It sustains environmental monitoring through pollutant detections.

All these valuable tasks are nowadays performed using either trained animals, or static and hand-held sensors. Search dogs are extremely efficient in using their olfactory sense to locate the object of interest. For instance, avalanche dogs can cover more ground in less time than several humans do. However, their ability depends on their training which is expensive and time consuming. Moreover, a human handler needs to follow the search dog in its exploration. This is also the inconvenience of hand-held sensors. Static sensors, on the other hand, are mostly autonomous and inexpensive, but a single sensor can only cover a single point in space.

For a better coverage a sensor network is required, which becomes proportionally expensive and energy consuming while leaving dead zones in between the sensors. In addition, complex scenarios require a consistent framework where all the measurements are integrated and interpreted, which might be too difficult for search dogs and humans with hand-held sensors.

Autonomous mobile robots capable of localizing the source of an airborne chemical will bring major improvements in performance and convenience to the applications in different domains.

Since robots are less affected by distractions in the environment, they are more robust than living beings in critical conditions. Cost-wise, the production and maintenance of a robot is significantly less expensive than managing and training a search dog, which takes years and the owner needs to live with the dog to properly understand it. In terms of the training for the user, in case of a robot, reading a user manual would suffice, while for a dog, the handler would need to complete a training to be able to accompany a search dog in action. Additionally, in case of a technical problem or an accident, which is not unforeseeable in harsh conditions, robots are more easily replaceable.

Furthermore, robots, when programmed accordingly, are supposed to accomplish their missions without human intervention. Thus, they should be able to accomplish their mission on dangerous zones where humans and dogs cannot set foot. Moreover, when more than one robot is available, they can be grouped in a team and leverage communication to obtain more information faster than a single robot and locate the source of interest in a shorter time.

For such a system to be functional in real environments, multiple aspects need to be addressed:

- Mobility: the system should be able to move appropriately on the targeted field.
- Sensing: on-board sensors need to be sensitive to the substance released by the sought source.
- Algorithm: the software must leverage the sensing data and the system's mobility efficiently to find the source in a timely fashion.

Regarding the mobility, the robot needs to be agile and robust to the considered field conditions to carry out the task seamlessly. Nowadays, robust ground robots exist that move as reliably as a living dog (e.g., Spot from Boston Dynamics) or a living spider (e.g., T8X from Robugtix). In terms of flying vehicles, small and agile quadrotors exist that are both indoor and outdoor compatible and could be used in such scenarios (e.g., Elios from Flyability and Crazyflie from Bitcraze).

Unlike the research progress on robot mobility that is progressing fast toward matching the one of some living beings, the performance of today's gas sensing technology still cannot be compared to natural olfactory sensors. Accurate and fast gas sensors lack portability and are not cost effective, and therefore cannot be mounted on mobile robots. Small sensors, on the

other hand, have a slow response time and low accuracy in discriminating well a particular gas.

Leveraging novel and intelligent algorithms, however, makes it possible to make use of imperfect sensory data to locate the source of interest. The algorithm deployed on the system needs therefore to make up for the inaccuracy of the acquired data using data processing methods. That is why research on gas source localization algorithms and methods has been attracting a considerable attention from roboticists and environmental engineers in the past thirty years.

1.2 Challenges

Despite the growing attention to the topic of Gas Source Localization (GSL) in the research community, this critical task remains a challenge. The main difficulty arises from the gas dispersion phenomenon [2] which depends on many environmental factors.

The propagation of gas molecules in the air is due to two different phenomena [3]. Firstly, the molecular diffusion drives gas molecules away from the source randomly in all directions. This happens even when no airflow is present, as it is mainly due to molecular motions, which results in a relatively smooth gradient of gas concentration around the source. Secondly, when airflow is present, molecules are not only manipulated by diffusion, but also transported by advection through the wind. Hence, the gas trail, which is called the plume, is mainly determined by the wind movements.

The airflow profile itself is highly impacted by the surrounding environment. In a laminar airflow, when no obstacle is present and the wind is blowing constantly and continuously, as the plume moves away from the source, the plume becomes wider and the gas concentration decreases. In a turbulent airflow, on the other hand, the plume will not maintain a well-characterized shape [4]. Thus, the gas plume is shaped not only by the characteristics of the source, but also by the airflow which, in turn, is influenced by the environment.

Additionally, despite the common belief, a gas plume is not characterized by a smooth and continuous gradient. In a small scale structure, it has shown to be made of fluctuating patches of concentration floating in the air [5]. The gas patches become diluted and less concentrated while traveling away from the source with the wind.

The main challenging aspect of GSL is therefore related to the complexity of airflow and plume structure. The airflow is hard to reliably measure and control, which makes experimenting with it a non trivial task. To carry out repeatable experiments, a specialized environment with a controlled airflow would be beneficial, which is not a common facility in research centers.

Furthermore, the limited sensing capabilities of current portable sensors causes an additional difficulty. This is usually compensated by novel algorithms and methods that use the sensor data in an intelligent fashion. Many algorithms, presented in different categories, have proposed methods for GSL in the literature. Solutions have been proposed in various

configurations, such as different robotics or simulation platforms and with different number of robots, ranging from single agents, to swarms of 20 units [6]. However, since the flow complexity depends on the experimental environment, the performance evaluation of each method is highly impacted by the evaluation setup. Moreover, the large majority of the studies are performed indoors, due to the high complexity of gas dispersal outdoors. In fact, as the air flow becomes more turbulence-dominated, the plume structure becomes more complex with gas patches of different concentration levels and sizes [7], which varies in time. Such outdoors conditions are not repeatable, making more difficult the reproduction of the experiments and the systematic performance evaluation of the mobile sensing system.

An ideal algorithm would work in any environment with any airflow profile, and would be able to perform on a variation of different types of sensing assets or a combination of them as a team, with cooperative behaviors. To reach this end, the research community has still a long way to go, but in this thesis, we have tried to take a step in that direction by presenting a powerful method and enriching it further with novel features to adapt to complex scenarios.

In the following chapter, we will visit an overview of the state of the art on GSL methods and algorithms, along with their potentials and limitations.

2 Related Works

To simplify the challenging problem of GSL, scientists usually divide it into three sub-tasks [8]:

1. Gas plume acquisition, which refers to a stage that aims to find the plume in the environment.
2. Gas plume tracking, that is the phase where the robot follows the samples obtained from the plume to localize the source.
3. Gas source declaration, the final task, during which the agent validates and declares the location of the source.

Even though there are algorithms capable of solving all three subtasks within a single method, without making a distinct separation between different stages, the sub-tasks can be seen as separate but sequential missions. In the literature, methods are presented that address GSL only partially, by solving only one or two of the sub-tasks, as each part uses and present different information.

The first subtask, for instance, can be seen as a purely search problem, where the robot needs to explore the environment to encounter the gas plume. The last one, on the other hand, depends highly on the application and the scenario and can leverage other sensing modalities than olfaction. Therefore, most of the studies in the literature focus on the plume tracking sub-task, which is the core and the most challenging part of the mission.

In this chapter, an overview of the leading methods in all three subtasks of GSL is presented.

2.1 Gas Plume Acquisition

The solution for GSL starts with finding the gas plume. In case no wind in present, this sub-task is usually completed using search or coverage algorithms, as the task is to explore the environment until the plume is found without using any cues. Besides deterministic search

methods with predefined trajectories, such as a raster scan, randomized search algorithms are among typical solutions in this case, such as Lévy Walks and Correlated Random Walks.

In 2009, a new search strategy for plume finding has been developed using the wind direction information [9]. Pasternak et al. designed a bio-inspired algorithm called Lévy Taxis which performs a random walk biased by the local wind direction.

Even though search algorithms such as Lévy Taxis have the potential to be extended to multi-robot systems, they are not yet done. One work which deeply studies the problem of gas plume finding with a group of robots is [10]. The focus of this work was to find an optimal swarm formation for gas plume acquisition using numerical optimization.

In this sub-task, it is hard to compare the performance of the existing techniques, because the difficulty of the search task depends on the ratio of the plume area divided by the total area [9]. Additionally, having multiple agents reduces the spent time and distance [8]. This implies that a multi-robot system is more efficient in the first phase compared to a single-agent algorithm.

2.2 Gas Plume Tracking

The second sub-task of gas source localization consists of tracking the gas plume in order to reach the source. It is considered as the part the most peculiar to chemical sensing in general and less depending on the application, this is why it has been the main focus of most of the studies in this field. Unlike the first phase, there is a large variety of strategies for gas plume tracking. For a better representation, we have divided the algorithms into different categories.

2.2.1 Gradient-based algorithms

The earliest works on gas source localization started in the 1990s. In these contributions the characteristics of gas dispersal were not considered. For instance, the work reported in [11] was based on the concentration gradient calculated using multiples samples taken at different positions. It has been followed by [12] using a simulated cooperative swarm of robots with the same strategy.

This type of algorithms, while being the most intuitive and computationally light ones, they perform well only in presence of low-noise gas concentration signal which in turn requires a relatively long time to find the source [12].

2.2.2 Bio-inspired algorithms

The most studied class of algorithms for the GSL problem, takes inspiration from living organisms such as moth, bacteria, dung beetle, etc. [13].

Since as early as 1996, moths' behavior has been imitated for robotic gas source localization

[14]. In [15] and [16] Lochmatter presented a novel moth-inspired algorithm called surge-cast, tested along with two other algorithms of the same class, namely casting and surge-spiral, with a single robot.

While theoretically the performance of a multi-robot system is expected to be better than that of a single robot, in [17] the performance of a non-cooperative multi-robot system, performing moth-inspired algorithms, was found to be similar to a single robot, for most configurations.

Moreover, in [18] moth-inspired algorithms are shown to be able to partially deal with obstacle-full environment using simple obstacle avoidance strategies.

Overall, bio-inspired algorithms have the advantage of requiring small memory and computational resources for the agent. Nevertheless, due to low performance of current sensing and locomotion technologies compared to their biological counterparts, these algorithms are still far from being reliable in realistic environments [19].

2.2.3 Metaheuristic algorithms

A number of uncertainties and unknown disturbances characterize a distributed sensing system engaged in a GSL task. As these uncertainties are not easily modeled, universal approximation methods such as neural networks can be used in order to estimate and therefore cancel the effect of uncertainties [20].

Particle Swarm Optimization (PSO) is a cooperative multi-agent algorithm, mainly used for multi-dimensional optimization problems. According to many researchers, the problem of searching gas source in an environment can also be seen as a stochastic optimization problem, where the goal is to find the local maxima in the average gas concentration map [21]. Although this algorithm usually presents an efficient search behavior, there is a high chance that the particles get trapped in a local optimum when the search space is complex, which is the case in an gas plume. As a result, in most of the works, the PSO algorithm has been modified or combined with other methods before being applied to the GSL problem.

For instance, Jatmiko et al. in [22] used a modified PSO method to localize the gas source in a simulated dynamic obstacle-filled environment with a meandering gas plume. In their work, a modified version of PSO algorithm was implemented for guiding autonomous robots towards the higher gas concentration. In several other works, such as [21], [23] and [24], different versions of modified PSO algorithms have been used as a control method for robotic swarms, successfully solving the problem of gas source localization.

In [25] Li et al. combined PSO algorithm with a probabilistic approach to guide a simulated swarm of robots towards the gas source. The entropy of the posterior probability distribution of source location acts as the fitness function which PSO tends to minimize.

Genetic Programming (GP) is an evolutionary technique inspired by natural evolution. In a

more recent work [3], GP is used as an optimization tool for training a mobile robot localizing the gas source. They define a set of possible actions for the robot, from which the algorithm evolves to create different behaviors. GP-based methods and many others mentioned below, could be considered as bio-inspired techniques and be classified as such, but since the behavior is not directly inspired from the nature, but rather the underlying method that draws and shapes the behavior, we prefer to keep them in this class.

Glowworm Swarm Optimization (GSO) is another swarm intelligence multimodal optimization algorithm. In this algorithm, agents emit a signal that contains information about their current state, like the light emitted by glowworms. Each agent is then probabilistically attracted by a brighter glowworm neighbor. This mechanism splits the swarm into subgroups, located at multiple optimum points of a multidimensional space. For GSL this algorithm is usually applied in case of multiple sources present in the environment. By means of this technique, [26] and [6] have been able to simultaneously localize multiple gas sources using a swarm of about 20 robots in simulation.

Artificial Neural Network (ANN) represent a further computational approach to the GSL problem. Recurrent connections enable the network to have a memory of previous outputs, which seems to be convenient for gas source localization.

In 1996, a hybrid system has been designed, combining living antenna of a male moth and a microrobot controlled by a simple recurrent neural network [27]. The electric signals from the living sensors were fed to an eight-neuron recurrent network in order to perform a moth-like behavior in response to the pheromone.

In [28], a simple chemotaxis behavior is first implemented in a six-node fully interconnected Continuous Time Recurrent Neural Network (CTRNN). Then its parameters are mapped onto a genetic string and are evolved using a Genetic Algorithm. Although, in 2013, de Croon et al. [29] used a similar ANN for the same purpose, but in a more complex environment.

All in all, as shown by different works, optimized controllers, despite their relatively complex implementations, have some advantages: first of all, while being more complex than two previous classes, they are generally computationally inexpensive, because they use very simple instructions or a small ANN. Secondly, the obtained controller results in a strategy that depend more on the context of the problem rather than on the designer's bias, which leads to robust strategies. Finally, they can be presented as a synthesis of multiple single methods, such as the case of moth behaviors, which is more closer to the natural model. Nevertheless, these algorithms need a learning stage that, besides being a time-consuming process, requires precise information about the environment and gas dispersion phenomenon to be competitive.

2.2.4 Formation-based algorithms

A relatively newer class of algorithm is based on agents formation in the environment. These algorithms have been natively designed for multi-robot systems. Hence, they can sample the

gas concentration in different positions at the same time, which is a significant advantage over algorithms that are limited to a single agent. All the robots share their observations (gas concentration, wind direction) with other members of the group and determine their relative position as well as absolute heading respect to the wind direction.

The robots formation has been exploited for GSL for the first time in [30]. The developed algorithm is called crosswind formation algorithm, and it involves keeping at least one robot on each side of the plume, while going upwind. The results of experiments with three and five real robots showed that this algorithm achieves close-to-optimal performance in simplified environmental conditions. The formation-based algorithms have been thoroughly studied in [31].

Recently, a 3D formation-based system has also been presented in [32]. In this work, they use a combination of ground and an emulated aerial robot (a mobile robot mounted on a traversing system inside a wind tunnel) in formation.

In general, these methods have low computational and memory requirements. However, since the coordination between agents is necessary, the method completely relies on inter-robot communication and relative positioning which could be challenging to maintain in a realistic environment.

2.2.5 Probabilistic algorithms

Researchers in GSL have been modeling the source location as a probability distribution which is derived from the observations made by the agent in the environment [33]. After each new observation, the probability distribution modeling the source location is updated using recursive Bayesian estimation. This process continues until the probability distribution of the source reduces to a Dirac function.

One of the main algorithms applying probabilistic approach to GSL is called Infotaxis [34]. In this algorithm, tracking the maximum amount of information will guide the robot to the source. This algorithm has been tested and compared with three reactive moth-inspired algorithms in [35]. In another study [36], Infotaxis turned out to be robust to environmental changes.

Another probabilistic method in the field of GSL is to use Bayesian Inference (BI) to update the belief on the source location. This algorithm has been developed and assessed in a turbulent flow containing a chemical plume by Pang et al. in [37].

In [38], the probability distribution of the gas source location is represented as a Particle Filter (PF). Li et al. tested this strategy in a time-variant air flow outdoor environment. In their work, at each iteration, the particles are placed in respect of the wind velocity perceived by the robot. Then at the position of each particle, the probability distribution of the source location is calculated.

Another widely used method is called Source Term Estimation (STE) [39] which relies on a plume model and aims to learn the parameters of the model while the sensory system gathers data in the environment. Since the concept is very broad, the algorithm is not exclusively used for gas sources; depending on the model, it can be applied to chemical, biological, radiological or nuclear substance as well (e.g., radiation [40]). Neither is it limited to mobile sensor nodes, as it can be used with a static sensor network (e.g., [41]).

Some other techniques include creating a map of the environment with corresponding gas concentration by combining the measured gas concentration with the agent's location estimation. Their goal can be either localizing the source or simply monitoring the gas distribution in the environment. One of the challenges of this approach is to chose the convenient path planning strategy in order to increase the map accuracy, while decreasing the energy consumption.

Farell et al. developed a plume mapping approach based on Hidden Markov Methods (HMM) in [42]. Their method provides information such as likelihood of source location as a function of position.

In another probabilistic method [43], Blanco et al. present a Kalman-filter-based approximation approach which estimates the concentration and the variances for each location. They use a sequential Bayesian estimation to generate a model of the gas distribution of the environment.

Kernel DM+V [44] is another approach on gas distribution mapping. It models the measurement variance in addition to the time-averaged distribution using a mobile robot that has to roughly cover the search area. In [45] a multi-robot gas distribution mapping algorithm has been presented using time series analysis.

Very recently, in [46] an exploration strategy for GSL was proposed which incorporates prior available domain knowledge about the gas dispersion process and the environment. A Bayesian interpretation of the advection-diffusion equations was used to estimate the source distribution.

In conclusion, despite being computationally more expensive than all other categories [33], probabilistic algorithms have several advantages. Firstly, unlike the previously mentioned methods that provide only the source position, they are able to present a richer set of information about the environment (e.g., [47]) or the source characteristics [39]. Secondly, because of the probabilistic nature of the methods, the final result is associated with a measure of uncertainty, which shows how trustworthy the data are. Finally, compared to the other categories, probabilistic algorithms are more flexible in terms of the type of underlying hardware (e.g., static, mobile, single or multi-agent).

2.3 Gas Source Declaration

Before declaring success, an agent has to make sure that the found position corresponds exactly to the gas source. Usually, to make the problem easier, it is assumed that the source is located in an area of a higher concentration compared to a given threshold. However, in many studies, such as [42] and [44], it has been shown that the source is not always at the position where the gas concentration is the highest. On the other hand, concentration variance was found to be significantly higher close to the source compared to other places [33].

In probabilistic algorithms, such as Infotaxis [34], involving the entropy computation of the source location, the source is declared when a position reaches a certain level of entropy.

Divergence is also shown to be a useful operator to indicate the source location in [48]. Thus, a point with a positive value for the divergence is considered as the source location.

As the source declaration depends very much on the application, some researcher proposed a vision-based approach to correctly identify the gas source. For instance, in [49], a combination of olfaction and vision has been used to identify the source.

3 Objectives and Outline

The main objective of this thesis is to take a step toward addressing the problem of GSL through an algorithm capable of adapting to complex scenarios.

Given the above mentioned advantages of probabilistic algorithms, we chose to focus on a method of this category. The only drawback of this type of algorithm is their high computation requirements. However, with the current progress rate of technology in producing pocket-size powerful processing units, the computational cost becomes quickly negligible on modern hardware, including the wheeled-robot on which we run the algorithm in this work.

Among all state-of-the-art methods of probabilistic algorithms, STE appears to be a complete and reliable framework. It is complete because, when coupled with a belief building method and an information-gain oriented navigation technique, STE can solve all three subtasks of GSL in one stage. It does not need to address plume acquisition separately, because every sample that does not represent a high gas concentration shows the plausible locations where the source is not located, therefore the robot is guided toward other points in space. It also takes advantage of the uncertainty level in source location belief to declare the source position.

Different components of a STE algorithm can be modified or enriched to adapt to complex scenarios. Moreover, its versatility in terms of sensing assets makes it possible to deploy on a multi-robot system with cooperative behaviors.

This thesis, therefore, is laid out in six parts. A brief description of each part and its corresponding objectives can be found in the following.

Part I - Introduction: In this part, we first introduce the motivation and applications of GSL in robotics while mentioning the most substantial challenges on the path of the researchers in this field. Then we present a literature review on the most relevant research works while making a classification of the different categories of methods. Finally, the layout and the main objectives of each part of this manuscript are presented.

Part II - Platforms and Tools: In this part, we introduce the facility, the robotic platform

and the main hardware and software tools used in this work. Furthermore, we present the high-fidelity simulation tool adopted in this research and calibrated to the physical setup reproduced in the experimental facility.

Part III - Source Term Estimation Algorithm: In this part, the essence of the presented STE method is introduced with technical details. Then the results of baseline experiments are also presented and discussed.

Part IV - STE Algorithms for Complex Scenarios: In this part, the main STE method is gradually adapted to different complex scenarios. In each chapter, a new challenge is presented and solved by enriching the main STE method with a feature suiting the complexity of the scenario. Each scenario is then validated through simulation and/or experimental evaluations.

Part V - Benchmarking the STE Algorithm: In this part, we compare the performance of our STE method with two sate-of-the-art methods and show the potentials and limits of our proposed algorithm.

Part VI - Conclusion: In this part, we summarize the outcome of this thesis and core contributions of this work. Moreover, we present promising directions for research in this field.

3.1 Research Contributions

The main contribution of this thesis is to develop and evaluate a STE algorithm and adapt it to complex scenarios along three main axes: spatial dimension, environment, sensing system. In the following we describe the works that contributed along one of these axes to the state of the art.

Source Term Estimation Algorithm: We present an algorithm based on source term estimation for gas source localization that is coupled with a navigation method based on partially observable Markov decision processes. We propose an innovative strategy to balance exploration and exploitation in navigation. The method has been evaluated systematically through high-fidelity simulations and in a wind tunnel emulating realistic conditions. The impact of multiple algorithmic and environmental parameters has been studied in the experiments. The relevant publications of this part include:

- F. Rahbar, A. Marjovi, and A. Martinoli, "An algorithm for odor source localization based on source term estimation," IEEE International Conference on Robotics and Automation (ICRA), Montreal, Canada, 2019, pp. 973–979.

Source Term Estimation Algorithm for Unknown Environments and for 3D search space: After extending the search space from 2D to 3D in simulation, we study two variants of the STE algorithm, one exploiting a global and the other one a local framework. The local framework can be leveraged when no map of the environment is available for navigation and

for estimating the source position in it. It also allows for reducing the accumulated traveled distance by the robot and thus save energy. Relevant publications include:

- J. Ruddick, A. Marjovi, F. Rahbar, and A. Martinoli, "Design and Performance Evaluation of an Infotaxis-Based Three-Dimensional Algorithm for Odor Source Localization," IEEE International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018, pp. 1413–1420.
- F. Rahbar, A. Marjovi, and A. Martinoli, "Design and performance evaluation of an algorithm based on source term estimation for odor source localization," MDPI Sensors, vol. 19, no. 3, 2019.

Source Term Estimation Algorithm for Distributed Systems: we extend the previously presented STE algorithm for homogeneous multi-robot systems. By gradually increasing the level of coordination among multiple mobile robots, we study the benefits of a distributed system on reducing the amount of time and resources necessary to achieve the task at hand. The performance of the presented method is thoroughly evaluated and presented for different levels of coordination between robots. Relevant publications include:

- F. Rahbar and A. Martinoli, "A Distributed Source Term Estimation Algorithm for Multi-Robot Systems," IEEE International Conference on Robotics and Automation (ICRA), Paris, France, pp. 5604–5610, 2020.

Source Term Estimation Algorithm for cluttered environments: After developing a data-driven gas plume model for cluttered environments using our calibrated simulation setup, we integrate it in our STE framework and systematically evaluate it. The effort of this paper will be soon submitted for publication.

Another relevant publication on information-driven gas distribution mapping for cluttered environments in collaboration with a visiting PhD student of the University of Málaga (Andres Gongora) is under revision and will be resubmitted soon.

Benchmarking the Source Term Estimation Algorithm: The proposed STE algorithm is compared in a 2D search space in terms of performances with a competitive bio-inspired algorithm called Adaptive Lévy Taxis (ALT) that achieves gas plume tracking through a correlated random walk. Moreover, a comparison between the performance of the 3D variant of the proposed algorithm and another 3D bio-inspired algorithm is presented. Relevant publications include:

- R. Emery, F. Rahbar, A. Marjovi, A. Martinoli, "Adaptive Lévy Taxis for Odor Source Localization in Realistic Environmental Conditions," IEEE International Conference on Robotics and Automation (ICRA), Singapore, 2017, pp. 3552–3559.

- F. Rahbar, A. Marjovi, P. Kibleur, and A. Martinoli, "A 3-D Bio-inspired Odor Source Localization and its Validation in Realistic Environmental Conditions," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, Canada, 2017, pp. 3983–3989.

3.1.1 Contributions of Collaborators

I would like to give credit to my close collaborators who influenced the path followed in this thesis.

Ali Marjovi, my research colleague, was involved in shaping the ideas used in the early stages of this thesis. He also contributed in writing several of the published papers which are also revisited in this manuscript.

This thesis includes the work of Romain Emery, a master student who designed and evaluated the Adapted Levy Taxis (ALT) algorithm, under Ali's and my supervision. The work of Pierre Kibleur in a semester project under Ali's and my supervision resulted in the 3D bio-inspired algorithm, which is also presented in this thesis.

I shared the experimental facility with one of my colleagues, Chiara Ercolani, who substantially contributed in the setup of the motion capture system and other materials in the wind tunnel.

Part II Platforms and Tools

4 Experimental Facility and Tools

Rigorous and reliable experiments are an essential part of a research project in robotics. In this work, we leveraged many state-of-the-art tools and an experimental facility to provide accurate performance evaluation for our algorithms. In this chapter, we present in detail the tools that were used throughout this project to design and evaluate the developed methods.

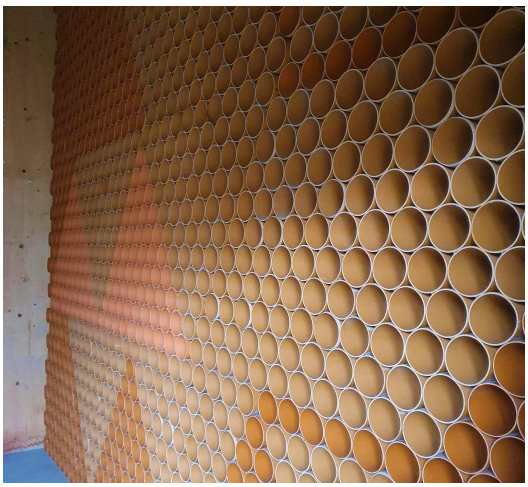
4.1 Wind Tunnel

To evaluate the performance of the algorithm in a repeatable fashion, our experiments were carried out in a wind channel of volume $18 \times 4 \times 1.9$ m^3, which provides a controllable wind flow. The wind is controlled by a fan placed outside of the experimental chamber. When activated, the fan generates an air flow that passes first through a flow straightener which consists of a set of narrow tubes of 10 cm in diameter, assembled in a honeycomb-like structure, as illustrated in Figure 4.1a. The resulting Reynolds numbers after the honeycomb would be between 4000 to 6000 for the considered wind speed range and a temperature of 15 °C. However, the flow gets further laminarized through a very fine-structured filter (see Figure 4.1b), right before entering the main chamber where experiments are carried out. Given the design of the wind tunnel and the experimentally observed data, we assumed that the resulting air flow is quasi-laminar.

The main chamber is a suitable environment to deploy robotics experiments. It is also equipped with devices necessary for measuring, monitoring or tracking. An overview of this experimental chamber with the deployed equipment is presented in Figure 4.2.

4.2 Khepera IV Robot

The robots used as an autonomous agent in all the experiments in this project is the Khepera IV robot, produced by K-Team. It is a compact differential wheeled robot with state of the art features (see Figure 4.3a).



(a) Flow straightener

(b) Inlet filter

Figure 4.1 – The two air filters of the wind tunnel.

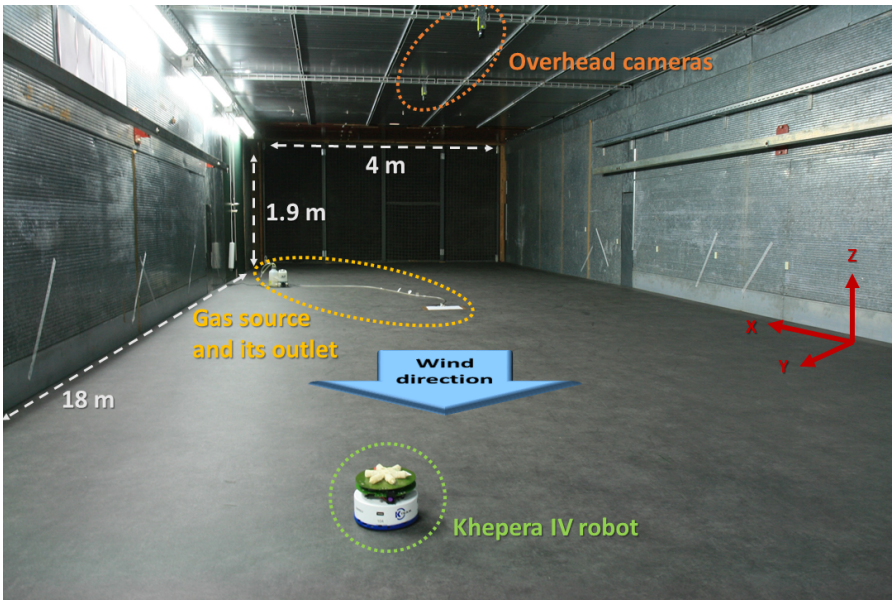
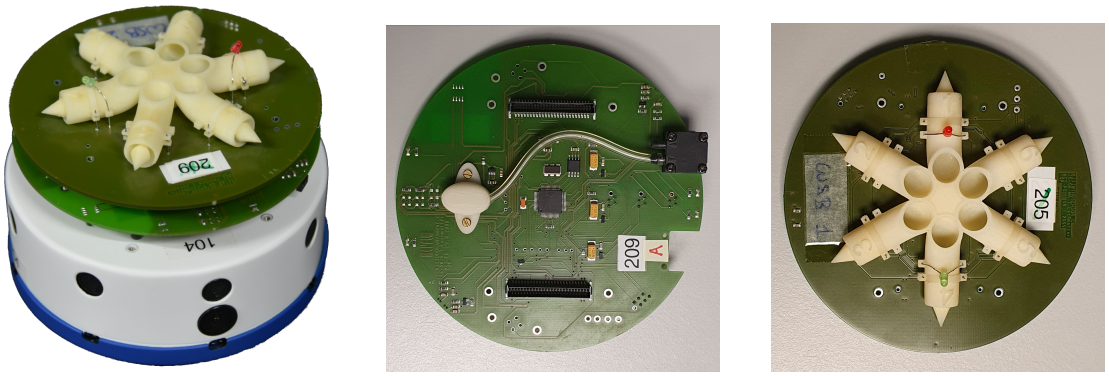


Figure 4.2 – A overview of the experimental facility, along with a Khepera IV robot, the gas source as well as the overhead cameras.



(a) Khepera IV robot (b) Gas sensor board (c) Wind sensor board

Figure 4.3 – The Khepera IV robot and its extension boards.

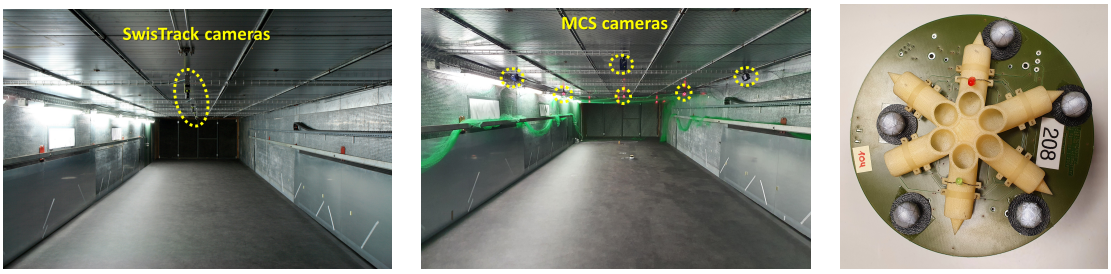
This robot is equipped with different types of sensors (e.g., ultrasound, accelerometer, gyroscope, camera, etc.) but we only use the infrared proximity sensors for collision avoidance. The eight infrared sensors are uniformly distributed around the robot, and have a detection range of 2 to 250 mm.

What makes this robot suitable for research is that it allows for custom made extension boards, for different purposes. For this project, we used two extension boards, shown in Figure 4.3, that were previously designed in DISAL for olfaction-related experiments [33]. One board is equipped with an olfaction sensor MiCS-5521 CO/VOC [50] as well as a pump that flows the air toward the sensor to accelerate the gas concentration measurements, as depicted in Figure 4.3b. According to experiments shown in [31], this sensor has a response time (to 90%) of 0.13 s and a recovery time (to 10%) of 0.41 s. In this thesis, the measurements are usually taken at 10 Hz and averaged over a 5 s window. In addition, the measurement error is studied and taken into account in the algorithm. The second extension board, shown in Figure 4.3c, is a custom made wind sensor that measures the angle of the wind direction.

The Khepera IV robot runs a Linux operating system (Angstrom) on board and allows for WiFi communication. Most of the codes, in this project, are written in C/C++ and cross-compiled on a desktop computer. Once the executable is transfered to the robot via WiFi, the robot runs the code autonomously on board. The WiFi communication is also used between robots to share information as well as for positioning systems.

4.3 Positioning Systems

In most of the experiments carried out in this project, an absolute positioning system is used. The position information, which is composed of coordinates (x, y) of the center of the robot along with its heading θ, was sent back to the robot, when needed, to be used in the algorithm. In all cases, the absolute position of the robot was logged and used as ground truth for performance evaluation and trajectory analysis.



(a) SwisTrack cameras (b) Motion capture system cameras (c) Infrared passive markers

Figure 4.4 – The two positioning systems in the wind tunnel.

4.3.1 SwisTrack

The majority of the experiments used the SwisTrack system [51] for tracking the robot's pose in the wind tunnel. In this system, which was previously developed in DISAL, color cameras and active markers were used. Six cameras were installed on the ceiling of the wind tunnel with overlapping field of views, as shown in Figure 4.4a. The active markers were a red and a green LED on top of the robot's extension board, placed with a fixed distance, as seen in Figure 4.3c. The open-source SwisTrack software localizes the robot by detecting the red and green LEDs in the cameras' feed and translating the pixel position to real-world coordinates using a previously done calibration. The position of the robot, i.e. the central point of the two LEDs, as well as the heading are both broadcasted on the network.

The accuracy of this system was below 10 cm which was acceptable for our experiments. However, for the active markers to be seen by the software, the ambient light had to be kept low.

4.3.2 Motion Capture System

In the later stages of this project, a motion capture system was deployed in the wind tunnel which allowed for a higher positioning accuracy and reliability. The system, provided by Motion Analysis Inc., is composed of 13 infrared cameras as well as a tracking software. Additionally, we placed reflective passive markers on top of the robots, as illustrated in Figure 4.4c. The placement of the markers had to be different for each robot in order for the software to identify them correctly.

This system allows for a sub-millimeteric accuracy and a tracking frequency of more than 100 Hz. The active infrared cameras did not interfere with the infrared sensors of the robots and the tracking could be achieved under suitable ambient light.

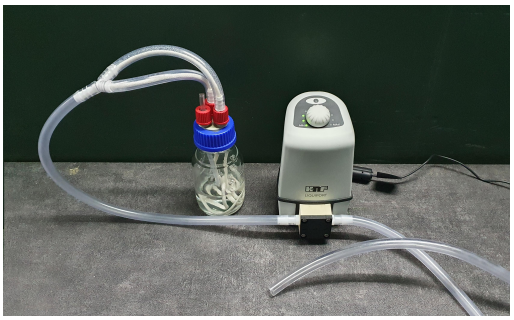


Figure 4.5 – Electric pump and ethanol recipient used as gas source.

Figure 4.6 – 3-axis traversing system with a Khepera IV robot on top.

4.4 Gas Source

The gas source was emulated using an electric pump vaporizing ethanol. The electric pump draws the air from its inlet that is connected to a bottle of ethanol with a tube. The ethanol bottle has three openings: one allows for a tube to go deep in the liquid which makes the air flow in the ethanol for better evaporation. The other two openings are connected to the inlet of the pump and contain a thick cotton thread that absorbs the liquid and allows for easier evaporation from its surface. Therefore, the air that the pump draws in contains evaporated ethanol. It then exits from the outlet, which is also connected to a tube. The end of the outlet tube is placed at the desired source position to create the plume (see Figure 4.5).

The pump's flow rate can be adjusted using a continuous control knob with discrete indicators. The flow rate ranges from 0.2 to 1.3 liter per minute. In the experiments of this project, we set the flow rate to less than one third of its capacity and is kept constant.

4.5 Traversing System

The wind tunnel is also equipped with a controllable 3-axis traversing system shown in Figure 4.6. By mounting a Khepera IV robot on top of it, it enables 3D movements for the wheeled robot. We mainly used this setup to scan the wind tunnel for ground truth purposes, but in a few cases, we used this combination to validate algorithms in a 3D space.

5 Simulators

Simulations allow for faster prototyping, easier evaluation, and better understanding of robots behavior. While validating the performance of algorithms, we usually start by evaluating them in simulation, where the gas plume is visible and the behavior of the robot can be judged more precisely. In this chapter, we will present the simulators that we leveraged in this work and explain our procedure to gather data.

5.1 Webots

For all the simulation experiments, we used Webots [52], which is an open-source high-fidelity robotics simulation software. The simulation environment was extended with a previously developed gas dispersion plugin [53] which allows for a reasonably realistic simulation of gas plume, based on the filament-based atmospheric dispersion model proposed in [2].

The Khepera IV robot, equipped with an olfaction and anemometer sensor, was also simulated in Webots, which is reasonably faithful to the real robot's characteristics. The communication between robots is also available in Webots which is designed as a simplified channel model with no package loss.

Another feature that we leverage in our simulations is a supervisor code that can control some properties of the simulation. For instance, before the start of each experiment, the positions of the source and the robots are randomly modified and the position of the source is stored in a file as ground truth. Furthermore, once the goal position of the robot is determined, it is communicated to the supervisor to place an indicator on the ground which makes it easier to follow the behavior of the robots.

We simulated an environment similar to our wind tunnel in terms of shape. Therefore, most of our simulations take place in a $20 \times 4 \times 2\,\mathrm{m}^3$ environment. In the later stages of the thesis, we decreased the used area to $10 \times 4 \times 2\,\mathrm{m}^3$ to exactly match the length of the usable area in the wind tunnel, which means the area covered by the positioning system.

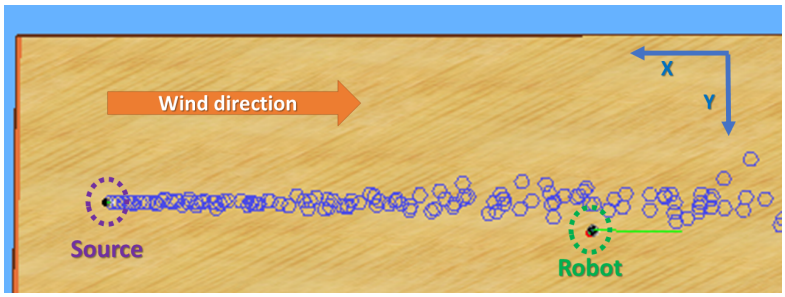


Figure 5.1 – Simulation environment in Webots, with the source upwind, the robot downwind, and the odor patches represented with blue hexagons.

The localization of the robot is done globally, using an emulated GNSS. When the robot does not use its global position in the algorithm, this value is kept as ground truth.

A view of the simulated environment can be seen in Figure 5.1.

In most of the experiments, the simulated wind flow is laminar and stationary in its intensity and direction (i.e., no meandering and wind gusts). In case obstacles were used in the setup, the wind flow needed to be simulated using OpenFOAM before being imported in Webots by the gas dispersion plugin. A brief explanation of our OpenFOAM setup is given in Section 5.2.

5.1.1 Calibration with wind-tunnel data

In order for the simulations to be a faithful representation of the real experiments, we calibrated the simulation environment as much as possible with the information coming from the real wind tunnel. The goal was to tune the parameters of the simulation environment in such a way that, in different environmental conditions, the plume (i.e. average concentration at each position relative to the source location) is as close as possible to the one generated in the wind tunnel.

Environmental conditions

The wind speed and source release rate are the two environmental conditions that we often vary in this project to evaluate the performance of algorithms in different situations. In the wind tunnel, these two parameters can be controlled by adjusting the speed of the wind tunnel fan and the power of the pump that acts as the gas source.

The speed of the fan is a value measured in rounds per minutes (rpm). In order to convert the fan rotation speed to wind speed in the wind tunnel, we need a function between the two. Therefore, for different fan speeds, we have measured and averaged the wind speed at the inlet, using a hand-held hot-wire anemometer. As the relationship seems to be linear, we fit a linear function between the two values. As a result, the wind speed in the wind tunnel can be approximated using the Equation 5.1. The fitted linear function and the measured values are

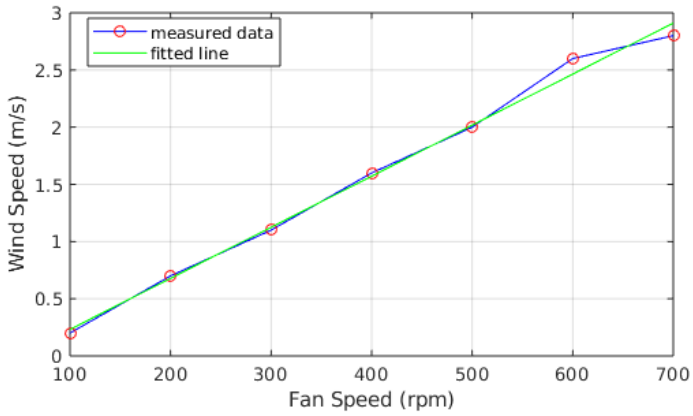


Figure 5.2 – Conversion of fan speed to wind speed in the wind tunnel.

shown in Figure 5.2.

$$speed_{wind} = 0.0044643 * speed_{fan} - 0.21429 \tag{5.1}$$

The other controllable parameter in the wind tunnel is the pump power which determines the source release rate. Since the exact value of released pure ethanol is not relevant in this thesis and also non-trivial to measure, we only consider the output power of the gas pump. As explained in Section 4.4, the power of the pump goes from 0.2 to 1.3 liter per minute and its control knob has 11 discrete indicators. Therefore, we assume that each indicator shows an increase of 0.1 l/min in the release rate. It is worth noting that the gas coming out of the source inlet is not pure ethanol, it is air mixed with evaporated ethanol. However, it is observed that by increasing the pump power, the evaporation of ethanol is also accelerated, and thus more ethanol is released.

Simulation parameters

In our simulation environment, and more specifically, in the gas dispersion plugin that simulates the plume, there are many parameters that need to be tuned for this calibration.

Besides the wind velocity and the source release rate, there are parameters related to gas filaments propagation, namely the amount of gas filaments, their width, their growth rate over time, and the standard deviation of diffusion. Since each of them affects the plume in a different way, we need their values to be correctly set to have the desired level of calibration.

Moreover, all the parameters related to the filament propagation should be fixed for all environmental conditions. The wind velocity can be set according to the Equation 5.1, and similarly, the release rate is assumed to be a linear function of the pump power. Therefore, there are six parameters to be tuned in total.

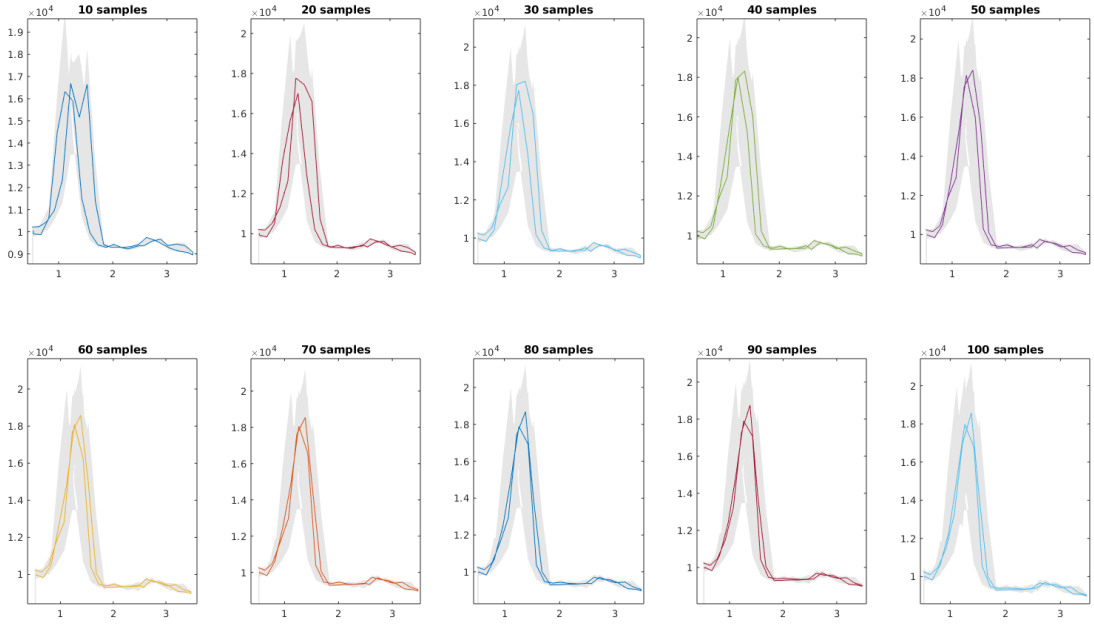


Figure 5.3 – Different number of samples and their average. The colored lines show the average values and the gray area, the actual measured samples.

Measurements and results

Since the reference for this calibration are the wind tunnel data, we first gathered data in the wind tunnel for different environmental conditions, i.e. different wind speeds and source release rates. For this purpose, we used a Khepera IV robot to scan an area of $10 \times 4\ m^2$, with a resolution of 15 cm in the Y-axis (crosswind direction) and 2 m in X-axis (upwind direction). The resolution is chosen to be higher in the Y-axis, as the concentration variation is higher in the Y-axis compared to the X-axis.

At each point, during this scan, the Khepera robot stops for 5 s to gather 50 gas concentration samples, (the sampling rate was 10 Hz). In order to make sure 50 samples give enough data for this calibration, for a given set of data points, we plot the average and the actual samples for 10 to 100 samples in Figure 5.3. Although slight differences exist between 50 samples and 100 samples, we can see that the general form of the signal is well presented. Therefore, for the rest of this thesis, we always keep 50 samples to calculate the averaged values.

Figure 5.4 shows the result of one of the scans in the wind tunnel, with a wind speed of 1 m/s and a release rate of 0.4 l/min. The light blue dots show the position on X- and Y-axes of samples. The Z-axis shows the concentration of each sample which is a relative value given by the VOC sensor. The vertical red line on each sample shows the standard deviation of concentration variations. As seen in this plot, the wind has a slight deviation with respect to X-axis, which also intensify with lower wind speeds. Since the deviation depends on the wind speed, we also included it in the simulated plume for a better accuracy of the calibration.

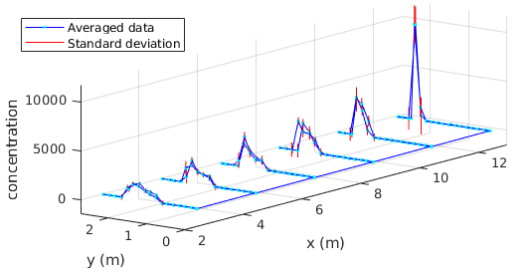


Figure 5.4 – The result of one of the scans in the wind tunnel.

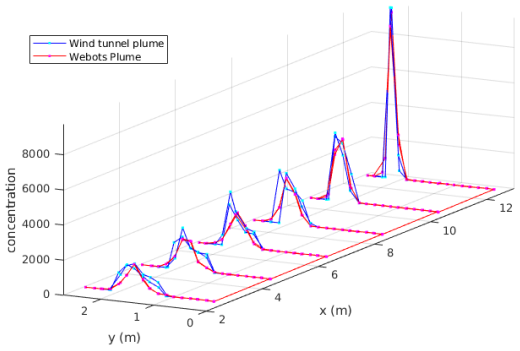


Figure 5.5 – The simulated plume obtained in Webots after calibration overlapped on the corresponding wind tunnel plume.

The metric used for this calibration is the Root Mean Squared Error (RMSE). We compare each sample point between the wind tunnel scan and the simulated plume and sum all the errors over all environmental conditions. Therefore, for each set of six parameters, we find an error value that we would like to minimize.

For this optimization problem, we used the Particle Swarm Optimization (PSO) method [54] from the Global Optimization Toolbox of Matlab. The objective function is designed to get six parameters and run a Webots simulation with the given parameters. In that simulation, emulated static sensors are placed exactly at the positions where samples are gathered in the wind tunnel to collect gas concentration. Then, the two plume are compared through RMSE.

Figure 5.5 shows a plume obtained in a Webots simulation with calibrated parameters, overlapped with the same plume shown in Figure 5.4. We can see that the width of the plume and the levels of concentration are very similar in both plots, while the shape of the simulated plume seems more regular and uniform.

Discussion and conclusion

The most prominent characteristics of olfaction-related research is the stochasticity of the measured field. Since many uncontrollable environmental conditions could affect the plume shape or the exact concentration levels, a perfect calibration with ideal accuracy is neither possible nor relevant. Even if it was possible to get a 100% accuracy in this calibration, it would not guarantee that another round of sampling would result in the exact same measurements. Therefore, in this calibration, we limited our expectation to similar plumes in terms of gas concentration, with consistency between different environmental conditions.

The wind tunnel provides a controlled environment, but it is still influenced by physical phenomenons that are out of our control. Here, we explain in a few points the main differences between the simulated environment and the one of the wind tunnel:

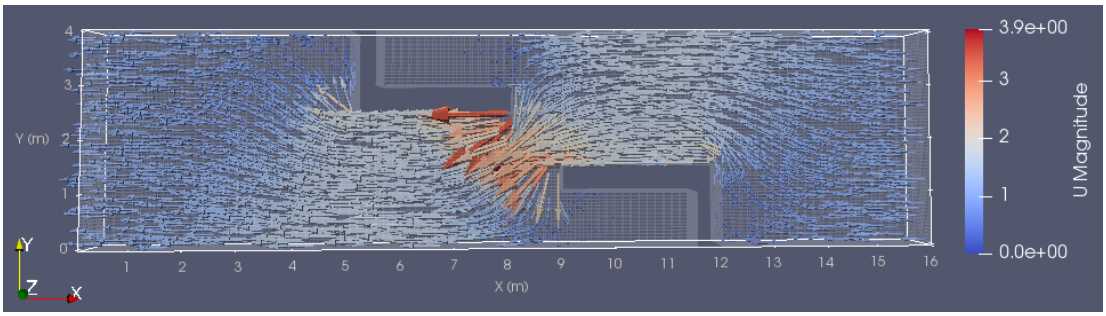
- Active source: the gas source used in the wind tunnel is active, while the one in the simulation is passive. An active source will cause a lower concentration at the very vicinity of the source compared to a passive one.
- Gas sensors: apart from the plume itself, the tool that measures the concentration is also different. The VOC sensors used in this thesis are calibrated only for ethanol concentration and might return slightly different values in different temperature and humidity conditions. Moreover, sensors can be damaged over time, which causes discrepancy between different sensors of the same type.
- Wind sensors: the simulated wind sensor receives the wind speed from the gas dispersion plugin, therefore its returned value is fully reliable. While it is possible to add noise to this sensor for a better representation of real wind sensors, it was kept at zero in our simulations.
- Baseline: in simulation, measuring the concentration outside of the plume will result in a value of zero. In wind tunnel experiments, since the sensors are not only sensitive to ethanol, their returned values outside of the plume are different from zero. This value, which can vary from one experiment to another, is subtracted from all the measurements.
- Gas residual: in simulation, each experiment starts with a clean environment where the source starts emitting gas. In long physical experiments, however, when ethanol was release in the wind tunnel for a long time, it is likely that traces of ethanol stays on surfaces outside the main airflow. These gas residual are then picked up by the gas sensor and impact the measurements.

In conclusion, given the complexity of the field that we tried to calibrate and all the uncontrollable and unmeasurable factors that played a role in shaping the plume, we believe to have achieved a satisfactory calibration between the simulated plume and the one in the wind tunnel.

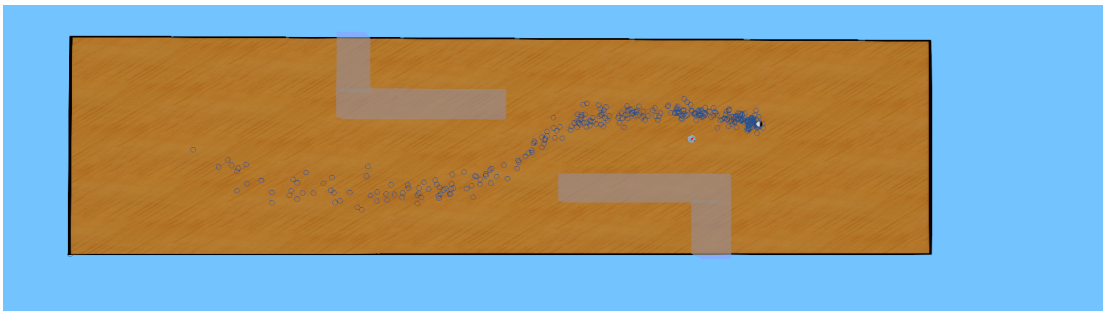
5.2 OpenFOAM

OpenFOAM is an open source CFD software with a wide range of applications for simulating complex fluid flows. In this thesis, we used OpenFOAM for simulating air flows when obstacles were present in the environment. The outcome of wind simulations were then used in Webots, through the gas dispersion plugin.

Different CFD algorithms are provided in OpenFOAM for different use cases, which are called *solvers*. The properties of the simulated fluid, the type of simulation, and the intended measurements are important factors in choosing the right solver. In our case, for a steady-state simulation of air flow around obstacles we used the *simpleFoam* solver which is designed for steady-state simulation of incompressible, turbulent flows.



(a)



(b)

Figure 5.6 – The air flow simulated in OpenFOAM (a) and the plume generated in Webots (b).

Therefore, after defining the same environment as in Webots, including the obstacles, a mesh is created in the entire volume. For every time step, determined by the time resolution, a log file is created with the wind velocity for every cell. This file is then loaded in the Webots simulation using the gas dispersion plugin. For a steady-state simulation, only one log file is used for all time steps, but in case of a turbulent-flow simulation, at each time step, the corresponding wind velocity needs to be loaded in the simulation.

The Figure 5.6 shows an example of simulated air flow in OpenFOAM and the resulting plume generated in Webots in a steady-state simulation with a wind speed of 1 m/s .

Source Term Estimation Algorithm Part III

6 Introduction

Source Term Estimation (STE) is a well-known inverse modeling technique for dispersal of hazardous chemical, biological, radiological or nuclear substances into the atmosphere. The spread of such material is modeled by atmospheric transport and dispersion models which require several variables as input, such as meteorological data, the strength and the location of the release. Sparse meteorological data are nowadays available in local weather stations. The strength and location of the release, however, are often unknown, and thus should be deduced from sensor measurements in combination with appropriate modeling techniques [39].

STE algorithms consist of a formal framework that aims to estimate the characteristics of a release source [39] by quantitatively estimating the parameters describing it. The nature of the parameters depend on the source and the way the release is represented in the algorithm, but in most cases, it includes the source location.

This framework usually involves different sub-methods and strategies to reach its final goal. The versatility of STE algorithms allows them to be performed on a large variety of sensing assets, ranging from static sensor networks [41] to mobile robots [55]. In cases where no mobility is involved in the system, the only goal of the method is to process the received data to estimate the sought characteristics of the plume. When deployed on robotic assets, however, the general structure of a STE algorithm will consist of two main parts of equal importance: estimation and navigation.

The estimation part, similarly to an immobile systems, leverages the observations to identify different characteristics of the plume. This part usually relies on a plume model whose parameters values are the information that we seek to estimate throughout the experiment using the data that the mobile sensing assets acquire. Two main approaches exist in the literature for estimation methods: optimization algorithms and probabilistic methods. The difference between the two approaches is that the output data in probabilistic algorithms is designed as a probability density function which is associated with a confidence level.

The optimization methods often rely on reducing the difference between predicted and

observed data in an objective function. To optimize the output of the objective function, a variety of difference methods have been leveraged, ranging from gradient descent methods [56, 57] to simulated annealing [58, 59] and genetic algorithm [60, 61].

Probabilistic approaches, however, are based on Bayesian methods that seek a Probability Density Function (PDF) on the source parameters which describes the parameters of the source as well as the uncertainty on them. The most probable values of the parameters are selected at the end as output. The PDF, deduced from the Bayesian method, is usually evaluated using stochastic sampling algorithms, such as Markov Chain Monte Carlo (MCMC) [41, 62] or Sequential Monte Carlo (SMC) [63, 64].

In the navigation part, the algorithm aims to maximize the amount of obtained information by sending the agent(s) to the most informative point(s) in the arena. Although the navigation methods coupled with STE are less tackled in the literature compared to the estimation methods, many typical strategies adopted in different application fields could be leveraged for solving the robotics navigation problem: autonomous search, informative path planning and control, multi-robot cooperation, etc.

In the literature, navigation strategies used with STE include pre-defined rules: for instance, the sensing agent moves in different directions with respect to the wind direction to collect samples [65, 66], or its motion is based on the expected information gain. The information-driven search strategy is often formulated as a Partially Observable Markov Decision Process (POMDP) [67] with different reward functions, such as Kullback-Lieber divergence [68], Rényi divergence [40], and mutual information [69].

The combination of an estimation and a navigation method represents a STE framework, as presented in Figure 6.1, which continues until the parameters are estimated with negligible uncertainty.

In terms of experimentations, the majority of the previous works applied STE either in simulation or on pre-recorded databases. Very recently, a few works in the literature deployed an STE algorithm on real mobile robots. The most prominent one [55], uses a coordinated multi-robot system and a particle filter algorithm to estimate the source parameters. The navigation is performed using a bio-inspired algorithm adapted to the probabilistic framework, and the method was evaluated in simulation and in physical experiments with a humid air plume.

In this part, we will present the STE algorithm developed in this thesis with all its components. Then we present the results of its performance evaluation to show the potentials of our proposed method. Further in this thesis, in Part V, we qualitatively and quantitatively compare this algorithm with other state-of-the-art algorithms.

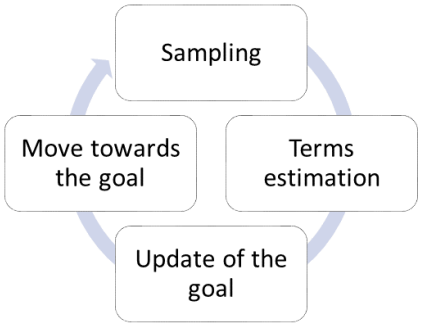


Figure 6.1 – The global structure of the presented STE algorithm

7 The STE Algorithm

STE algorithms are compilations of different components. Estimation and navigation are the most prominent parts of our proposed method, but there are other equally important elements that influence the proper functioning of the algorithm. A summary of different components as well as their relationships are presented in Figure 7.1.

In this chapter, we will describe in detail the theoretical and technical aspects of the proposed algorithm.

7.1 Plume Model

A plume model is often an analytical expression that defines the gas concentration for a given point in space, based on the source location and other characteristics such as source release rate, release start time, diffusion and advection parameters, etc. Probabilistic algorithms usually leverage a plume model for updating their belief about the source position by comparing the collected concentration data in the environment with the ones deduced from the plume model. Multiple definitions for the atmospheric dispersion model exist in the literature; they differ in terms of complexity, use-case, underlying environment, etc.

In this section, we describe the plume model that we used in our method.

7.1.1 Pseudo-Gaussian plume model

One of the most common plume models in the literature is the pseudo-Gaussian concentration plume model [70] which describes the time-averaged concentration model for a continuous point source in a laminar flow. It is presented in Eq. (7.1), where Q is the source release rate and $\bar{u}$ the average wind speed.

$$C(Q, x_s, y_s, z_s, \sigma_y, \sigma_z) = \frac{Q}{2\pi\bar{u}\sigma_y(x-x_s)\sigma_z(x-x_s)} e^{-\frac{(y-y_s)^2}{2\sigma_y^2(x-x_s)} - \frac{(z-z_s)^2}{2\sigma_z^2(x-x_s)}} \quad \forall x \geq x_s \tag{7.1}$$

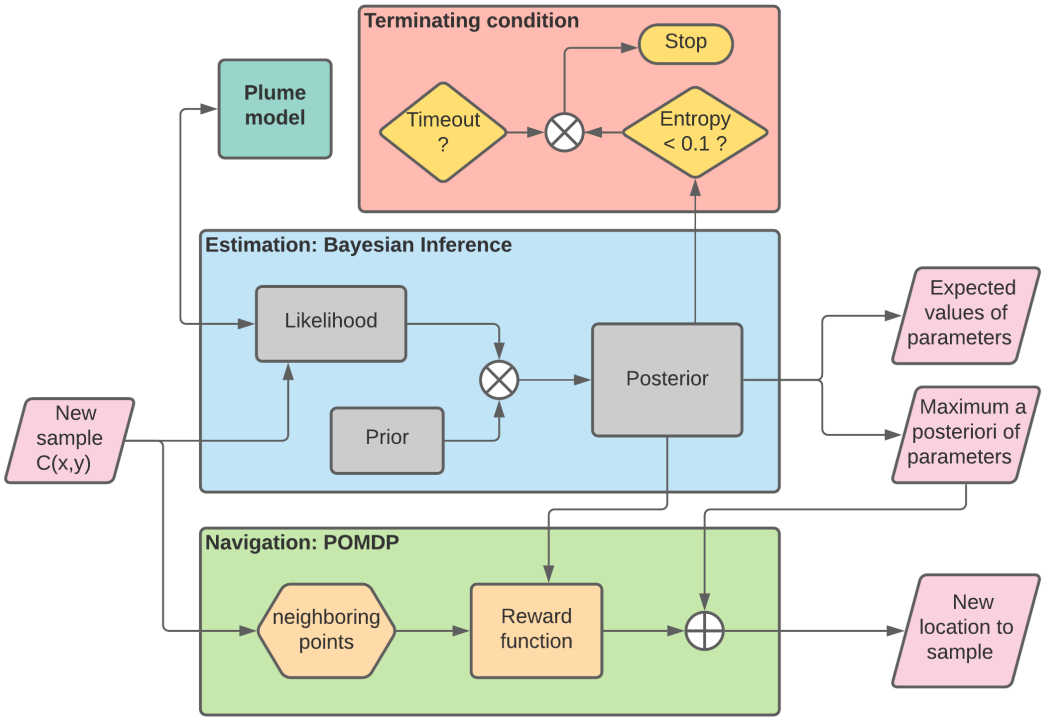


Figure 7.1 – The global structure of the presented STE algorithm

In this equation, the X-axis is assumed to be aligned with the direction of the airflow and it is defined for all the x downwind of the source (i.e., $\forall x \geq x_s$). The concentration is simply 0 for all points upwind of the source (i.e., $\forall x < x_s$). The source is positioned at (x_s, y_s, z_s) and σ_y and σ_z are the standard deviations of gas dispersion in the Y- and Z-axis, respectively; they are simplified to be linear functions of upwind distance from the source $(x - x_s)$.

Therefore, in a STE method, the goal is to estimate the set of parameters of the model, namely $m = \{Q, x_s, y_s, z_s, \sigma_y, \sigma_z\}$. As the standard deviations σ_y and σ_z are both of form $\sigma(x - x_s) = a(x - x_s) + b$, they each have two parameters to be estimated. Therefore, to simplify further, the plume is assumed isotropic in Y and Z directions, hence $\sigma_y = \sigma_z$. Thus, after the simplifications, six parameters remain to be estimated, making the problem six-dimensional. The range and resolution of all the parameters are chosen empirically.

When working in a 2D space, the equation is simplified as shown in Eq. (7.2) and the set of parameters to be estimated reduces to $m = \{Q, x_s, y_s, \sigma_y\}$, where the problem becomes five-dimensional.

$$C(Q, x_s, y_s, \sigma_y) = \frac{Q}{2\pi \bar{u} \sigma_y (x - x_s)} e^{-\frac{(y-y_s)^2}{2\sigma_y^2 (x-x_s)}} \quad \forall x \geq x_s \tag{7.2}$$

7.2 Parameter Estimation

Estimation is the most essential part of the algorithm, where the observations of the system are translated into belief about the parameters that we seek to estimate. In our proposed method, the estimation is performed probabilistically using the Bayesian formulation presented in Eq. (7.3), where m represents the set of model's parameters and D the obtained data through sampling. The posterior $P(m|D)$ represents the probability distribution on the parameters values.

$$P(m|D) = \frac{P(m)P(D|m)}{P(D)} \tag{7.3}$$

The evidence $P(D)$ being a normalization factor, it can be neglected. Hence,

$$P(m|D) \propto P(m)P(D|m) \tag{7.4}$$

We consider the prior $P(m)$ a uniform distribution in between the limits of each parameter. Therefore, the posterior $P(m|D)$ will be proportional to the likelihood $P(D|m)$ in the parameters limit, and equal to 0 outside.

The likelihood $P(D|m)$ defines the probability of obtaining a set of data, given a set of parameters. In other words, it returns the likelihood of a set of parameters given the data that the

robot has been collecting up to the present time. It is defined in [41] as follows:

$$P(D|m) \propto exp\left(-\frac{1}{2}\sum_i\left(\frac{(D_i - C_i(m))^2)}{\sigma_M^2 + \sigma_D^2}\right)\right) \tag{7.5}$$

where σ_M and σ_D represent the standard deviations of model and measurement error, respectively. Both errors are assumed to be normally distributed, with mean on 0.

The sum in Eq. (7.5) is applied on all samples D_i that the robot gathers during the experiment. $C_i(m)$ is the concentration determined by the plume model for a set of source parameters m for a sample i. It is defined using the plume model presented in Section 7.1. D_i is the actual measured data for the very same sample i. Since we use time-averaged plume models, the sample should be preferably represented by the mean of the sensed concentrations over a fixed time window. Therefore, to gather one sample, the robot stops at each target point for 5 s and then calculates the average of the sensed values gathered at 10 Hz (50 concentration values in total).

Since there are many parameters to estimate, the posterior probability density function has to be a multi-dimensional matrix, which is very time consuming to be entirely calculated. The solution to this problem would be to use an approximation algorithm such as the Markov Chain Monte Carlo (MCMC) [71] that allows evaluating the posterior probability function through efficient sampling.

In this work, we use the Metropolis–Hasting method [72]. The important factors in this method are the number of iterations and the proposal distribution which is chosen as a Gaussian function with the same dimensionality of the posterior probability function. Here, we only present the concept from a general perspective.

At each iteration, a candidate point p in the proposal distribution is chosen, which is equivalent of a set of parameters $p = \{Q_p, x_p, y_p, a_p, b_p\}$. Then, the likelihood of this point $P(p|D)$ is calculated and compared to the one of the previous point. The higher is the likelihood of the new point compared to the previous one, the higher is the chance of accepting it. In the case it is accepted, its likelihood is saved on the posterior probability distribution and replaces the previous point as the mean of the proposal distribution.

Once the posterior is calculated, we extract a few pieces of information from it, that are all relevant in the algorithm:

- Expected value: the average of the parameter sets, weighted by their posterior probabilities shows the current belief of the algorithm on the sought parameters. This value is essentially the output of the algorithm, i.e. when the algorithm ends, the source position is declared on the expected value of the posterior on each coordinate of the source position.
- Maximum a posteriori: the set of parameter with the highest probability could lead the

robot to a highly informative point, as it either radically changes the belief or reinforces it. Thus, this value is used in the navigation part, explained in Section 7.3.

- Entropy: the uncertainty on the posterior shows whether the belief is based on sufficient information. The entropy is, therefore, used to determine if the goal of the algorithm is reached and if so, the algorithm can stop. The ending strategy is explained in Section 7.4.

With the three outputs of the estimation part, the algorithm can continue the cycle by determining a new location in the search space to sample. This is done in the navigation part.

7.3 Navigation

The main goal of the navigation strategy is to feed the estimation part with worthwhile information that lead to an as precise and as quick as possible localization of the source. Therefore, it needs to predict which neighboring point is the richest one in terms of information, in order to send the robot to it. For this purpose, we propose to use POMDP [67] as a predictive navigation method to guide the robot in a profitable fashion.

POMDP requires three components for its operation: a posterior distribution, a set of possible actions, and a reward function. The posterior distribution is already given by the estimation part. As for the set of actions, to simplify the robot's motion and to reduce the computational cost, we limit the movement on one axis at a time. Since the movement is possible in both positive and negative directions on all three axes, the set of possible actions becomes six-fold in a 3-D environment. The most essential component of this navigation method is the reward function, as it determines the behavior of the robot. In this work, it is chosen to be the relative entropy (also known as Kullback–Leibler divergence) [73] which represents the gain of information from one probability density function P to another Q, defined in Equation (7.6). In our case, P would be the probability density function obtained through the estimation part, and Q the predicted probability density function that is calculated separately for each of the target points using the expected value of the current belief on the source parameters.

$$D_{KL}(P||Q) = \sum_j P(j) log \frac{P(j)}{Q(j)} \tag{7.6}$$

Thus, at each step, once the six possible target points are determined, the robot will first predict the concentration that would be measured in each of them, using the current estimation of the model parameters. Then, it calculates the potential update of the posterior probability function using the Bayesian formulation, which would result in the predicted posterior probability function Q. Finally, it evaluates the gain of information at each point using Equation (7.6).

The chosen target point leads the robot to the direction with the highest expected gain of

information. However, in the case where the robot does not have any information about the plume, i.e., no concentration is sensed, no direction would give more information than others. Hence, the robot tends to serially sample in all directions, and, as a result, it stays in the same region for a long time. To make the search more global, we need a component in the navigation strategy that promotes more exploration at the beginning, as well as leads the robot gradually towards the source when the estimation becomes more accurate. This indicating index can be seen in the evolution of the maximum a posteriori value of the source position.

Indeed, while the value of entropy is around maximum (i.e., very little information is available), the maximum a posteriori value, remains very unstable, but it converges towards the ground truth, similarly to the expected value, when the estimation becomes more accurate.

Therefore, we suggest to define the movement vector as a weighted sum of two 3-D vectors: the one leading towards the target point that provides more information, given by Kullback–Leibler divergence calculation, $\vec{V}_{KLD}$ and the vector that leads to the maximum a posteriori value of the source position $\vec{V}_{source}$.

$$\vec{V} = \alpha \vec{V}_{KLD} + (1-\alpha)\vec{V}_{source} \tag{7.7}$$

Intuitively, one would suggest that the algorithm should lead the robot towards the expected source position, as opposed to the maximum a posteriori. However, the expected source position never promotes exploration and might entrap the robot in an equilibrium. The coefficient α in Equation (13.1) determines how global or local the search would be. Its optimal value is studied in Section 8.1.1.

In later stages of the thesis, with the intention of reducing the traveled distance by the robot, we also take into account the distance of candidate points from the current robot's location. The detail of this improvement on the navigation method is explained in Section 13.2.

Once the new goal position is determined, the robot moves toward it to gather a new observation. Thus, the cycle continues with estimation, until the ending condition is met.

7.4 End of Algorithm

The algorithm stops when the uncertainty on the system's belief on the source parameter estimation becomes very low. More concretely, the uncertainty indicator is the entropy of the posterior probability function. When it goes below a certain threshold the algorithm supposes to have all the information it needs to declare the source position.

The threshold is chosen arbitrarily and is fixed for all experiment to about 2% of the maximum entropy of the posterior density function. One could set the threshold to 0 to ensure that the algorithm continues until the posterior density becomes a Dirac function. However, setting

the threshold to a slightly higher value, as we did in this work, allows for stopping earlier when the uncertainty becomes negligible.

Additionally, there is also a timeout that forces the algorithm to stop when the estimation takes more time than expected which is represented by a pre-established total number of iterations.

When one of the ending conditions is met, the algorithm will declare the expected value of the posterior as the position of the source along with other parameters of the plume model.

8 Performance Evaluation

To evaluate our method, we first carry out systematic experiments in simulation to study the impact of some algorithmic parameters. Then the same procedure was followed in experiments achieved in our wind tunnel facility to validate the performance in physical reality. Moreover, the impact of environmental conditions have also been studied. Finally, our proposed method was compared in terms of performance with another algorithm as benchmark. In the present chapter, we explain the experimental procedures in detail, as well as their outcome.

8.1 Simulation Experiments

Our STE algorithm is first evaluated in a 2D simulated environment leveraging Webots, as presented in Section 5.1. The simulated wind flow, in this set of experiments, is kept quasi-laminar and stationary in its intensity and direction (i.e., no meandering and wind gusts). To make the performance evaluation as fair as possible, we randomly set the initial position of the robot in the whole environment, as well as the position of the source on the Y-axis. On X-axis, the source remains on extreme upwind direction, with 1 m distance from the wall. This choice ensures that the setup is as challenging as possible while the flow condition remains uncomplicated.

Figure 8.1 shows a sample trajectory of the robot. In this example, the robot started far from the source. At the beginning of the experiment, large steps were taken to explore the environment. When the uncertainty gradually decreased, the maximum a posteriori value converged to the expected value and as a result the steps became smaller. Finally, for the uncertainty to become negligible, the robot sampled several points around the source position area before stopping the algorithm.

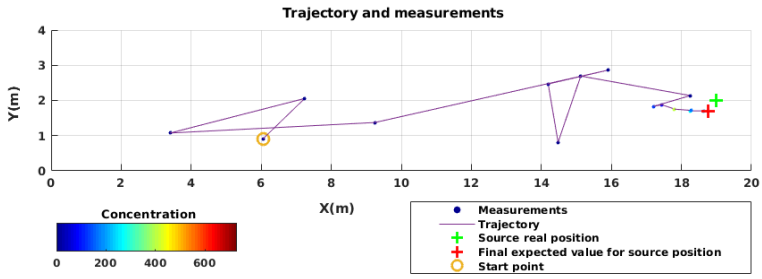


Figure 8.1 – Example of robot's trajectory in simulation. The initial position of the robot, as well as the real position of the source and its predicted one are shown. On each sampled point, the measured concentration is indicated by the color of the circle, which corresponds to the averaged value of the measurements in a 5-s time window. The measured concentrations are relative values given by the sensor in mV.

8.1.1 Algorithmic parameters

The presented algorithm has three essential parameters that impact the performance: model and measurement errors considered in the likelihood, σ_M and σ_D in Equation (7.5), that influence the uncertainty on the estimation, and the ratio α between exploration and exploitation in the navigation part. In the simulation, we assumed to have no measurement error (i.e., $\sigma_D = 0$), firstly, because in our simulation we could set the measurement noise to 0 and secondly, because this allowed us to isolate and tune the model error σ_M in our setup. The value of measurement error σ_D was studied later in physical experiments, as presented in Section 8.2. In the simulation, we evaluated the performance of the algorithm using different values of the two remaining parameters (α and σ_M) to find the best set of values.

The metric that we used for evaluation is the traveled distance before the robot reaches the end of the algorithm as well as the estimation error on the source position on the X and Y-axis. The algorithm estimates other parameters of the model as well, namely Q, σ_y and σ_z, but their values are not of primary interest for us as we focused on the performance of the robot as opposed to the faithfulness of the model to the physical transport phenomenon. Both metrics deliver positive values, with the lower the value, the better the performance. For each set of values, we repeated the simulation 10 times in a fixed environmental condition, where the wind speed was set to 0.9 m/s and the source release rate to 5%. This condition is labeled C in Table 8.1 and the actual values are mentioned in Table 8.2.

The results are presented in Figure 8.2. On each box, the central mark indicates the median, and the bottom and top edges of the box indicate the 25th and 75th percentiles of the results, respectively. The whiskers extend to the most extreme data points without considering the outliers, which are plotted individually using the "+" symbol. The parameter values corresponding to the labels are presented in Table 8.3. The results show that the likelihood model error σ_M has a direct impact on the traveled distance. Naturally, the more uncertainty is considered, the more time the algorithm needs to converge. However, too low values also have

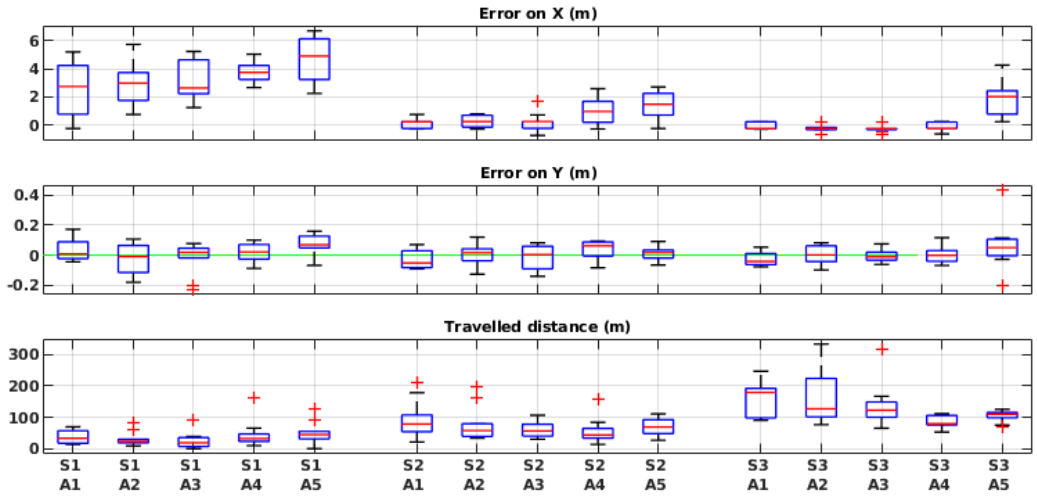


Figure 8.2 – Evaluation results in simulation with different algorithmic parameters in the global framework. See Table 8.3 for the meaning of the labels.

Table 8.1 – Environmental condition setups.

Label	**A**	**B**	**C**	**D**
Wind Speed	Low	Low	High	High
Release Rate	Low	High	Low	High

negative impact on the estimation of the X value of the source position. The justification is that the pseudo-Gaussian plume model that we use in our algorithm does not entirely match the model used in the simulation odor dispersion plugin. Therefore, we need to allow for some values of model error. Given the results, $\sigma_M = 30$ seems to be the best compromise between the two metrics.

As for the exploration–exploitation ratio α, when at the highest value $A5$ (high exploration), it visibly perturbs the accuracy of the algorithm. On lower values ($A1$–$A4$), on the other hand, it appears to affect mostly the traveled distance, but very slightly. Therefore, we chose the middle value 0.5, which means 50% ratio between exploration and exploitation.

Table 8.2 – Environmental parameters studied in simulations and wind tunnel experiments.

Parameter	**Tested Values**
Wind speed (m/s)	Simulation: 0.2 (low), 0.9 (high) Wind tunnel: 0.6 (low), 1.0 (high)
Source release rate	Simulations: 5% (low), 10% (high) Wind tunnel: 70% (low) and 100% (high)

Table 8.3 – Labels associated with algorithmic parameters value.

Label	S1	S2	S3
σ_M	10	30	50

Label	Sd1	Sd2	Sd3
σ_D	100	125	150

Label	A1	A2	A3	A4	A5
α	0.1	0.3	0.5	0.7	0.9

8.1.2 Environmental parameters

Once the best algorithmic parameters were deduced, we studied the performance of the method in different environmental conditions. We considered two environmental parameters that have a high impact on the complexity of the problem: wind speed and source release rate. Indeed, when the wind speed is high, the plume is narrow, i.e., the crosswind section of the plume is small. Therefore, once the plume is found by the robot, following it towards the source is relatively easy. When the wind speed is low, on the other hand, the plume takes a less defined shape, its crosswind section becomes larger, and the robot has a higher chance of failing in its task. Additionally, for a given source release rate, when the wind speed is low, due to diffusion, the gas becomes more dispersed in the environment and thus the concentration becomes lower at a given point located at the center of the plume axis, compared to a situation where the wind speed is high. Therefore, the contrast between two points that are inside and outside of the plume becomes less significant in low wind conditions. Furthermore, since the source release rate impacts the intensity of the concentration levels that the robot senses, the robustness of the method to the amplitude of the odor concentration data was evaluated by varying it. Table 8.2 lists the values of the environmental parameters in our experiments. Each set of experiments was repeated ten times.

The results of these evaluations are presented in Figure 8.3. The error is mostly very low on the source position in all environmental conditions. In the worst cases, it is around 10 or 20 cm on the Y-axis. On the X-axis, on the other hand, it is usually around 1 m. We suspect that the issue could come from the granularity for estimating the posterior probability density function, which is 0.5 m on the X-axis, while 0.2 m on the Y-axis. From a general point of view, however, we can see that the performance of the algorithm is very similar in all the tested environmental conditions. This result was expected, since the underlying model of the experiment adapts itself to both wind speed, which is captured using an anemometer, and release rate, which is estimated during the experiment by the algorithm.

The only cases where we observe a difference in performance, is in the traveled distance in condition A and the accuracy on X-axis in condition D. The higher traveled distance in condition A seems to be logical given the difficulty of the setup. In fact, the low wind speed does not give the plume a well defined shape and the low source release rate makes it hard for the robot to find cues for the estimation. Therefore, the robot needs to explore more in the

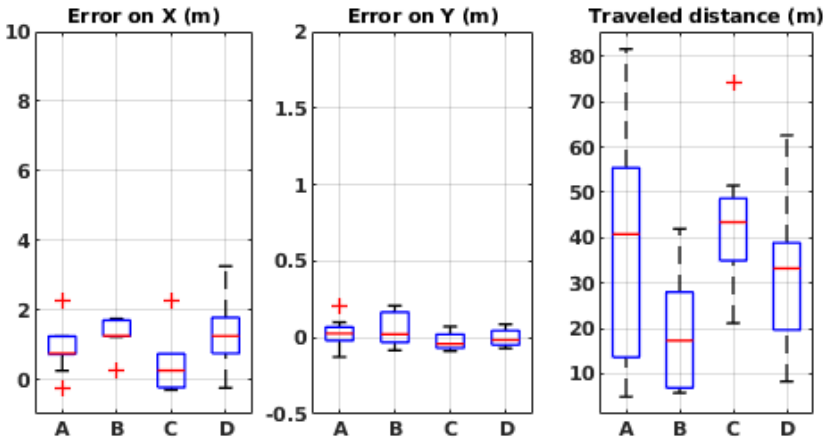


Figure 8.3 – Evaluation results in simulation for different environmental conditions in the global framework. See Table 8.1 for the meaning of the labels.

environment to collect valuable data and localize the source. The lower accuracy in condition D, on the other hand, could be caused either by the fact that the algorithmic parameters were studied and adjusted only in condition C, or by the discrepancy between the plume model (pseudo-Gaussian and representing the stationary distribution of the plume) used in the algorithm and the one used in the simulation (filament-based [2] and emulating the dynamics of the plume). Such discrepancy is in any case small as demonstrated in [2] and our implementation. Indeed, in [2], it is shown that the filament-based model matches the long-term time-averaged pseudo-Gaussian plume model. However, the comparison was made with a three minutes time-averaged data, and the chosen σ_y was a nonlinear function in the x coordinate. In our simulation setup up, the long-term equivalence of the two models was validated with a second degree polynomial function for σ_y. However, for the sake of simplicity, and since the coefficient of the second degree term was negligible, we chose to approximate the equations to a first degree linear function.

8.2 Physical Experiments

Figure 8.4 shows a sample trajectory of the robot in the wind tunnel. Similarly to the simulation experiments, first, the robot started to explore the environment by taking large steps. Once it found the plume and had less uncertainty about the source parameters, the steps became smaller and the overall movement was oriented towards the source. The final expected source position was only a few centimeter away from the real source position.

The concentration values show that the plume is very narrow close to the source, and therefore, a slight deviation from the plume axis results in a lower concentration. Additionally, there is a slight shift in the wind direction in the test environment, which is why the plume is not exactly aligned with X-axis.

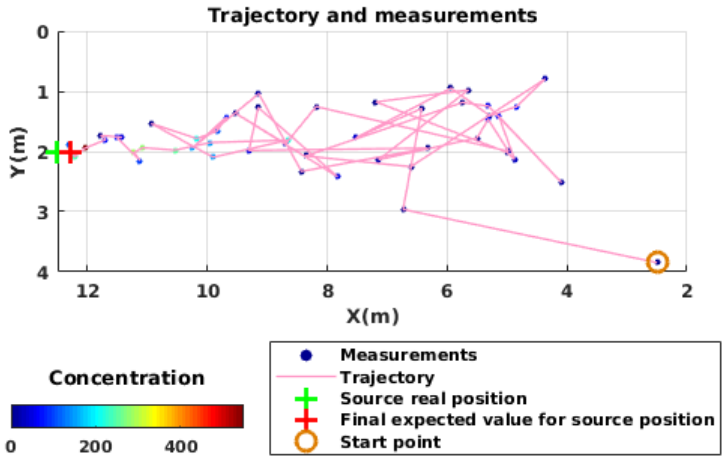


Figure 8.4 – Example of robot's trajectory in physical reality. The initial position of the robot, as well as the real position of the source and its predicted one are shown. On each sampled point, the measured concentration is indicated by the color of the circle, which corresponds to the averaged value of the measurements in a 5-s time window. The measured concentrations are relative values given by the sensor in mV.

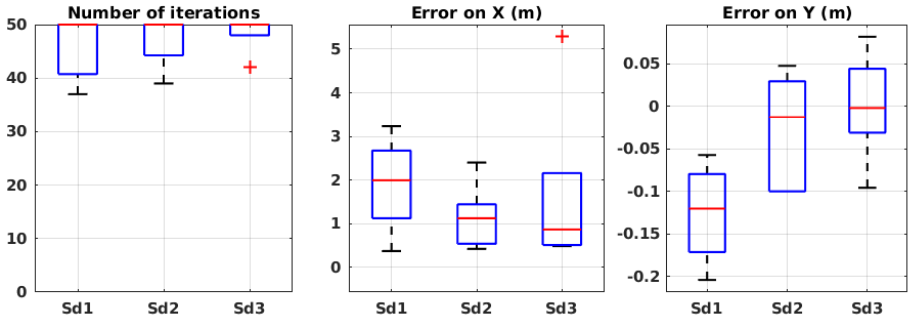


Figure 8.5 – Evaluation results in wind tunnel experiments with different values of σ_D. See Table 8.3 for the meaning of the labels.

8.2.1 Algorithmic parameters

Unlike in simulation, in physical experiments, it is inevitable to have measurement errors. Therefore, in our wind tunnel experiments, we studied the parameter σ_D of Equation (7.5), which is the standard deviation of the measurement error. In these experiments, the parameters σ_M and α were fixed to the best values deduced from the simulation results presented in Section 8.1.1.

Figure 8.5 shows the performance of the algorithm with different values of σ_D. In these experiments, the maximum number of iterations was fixed to 50. With $\sigma_D = 150$, the error on X and Y are very small, but the number of iterations is very close to 50. When $\sigma_D = 100$ the error on X is about 2 m on average, which is relatively high. This is why the middle value, 125, was chosen for the rest of the experiments.

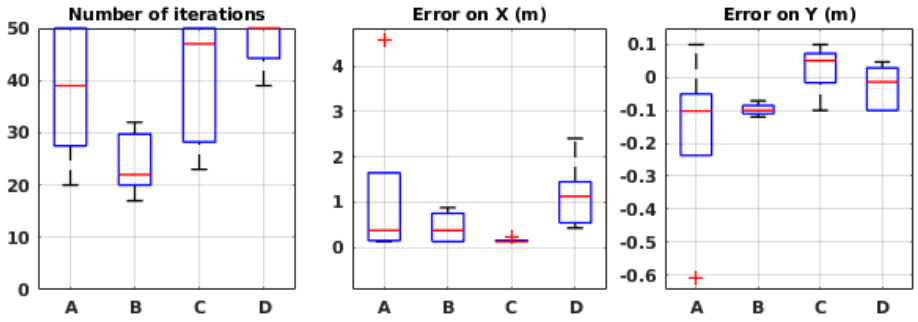


Figure 8.6 – Evaluation results in wind tunnel experiments in different environmental conditions. Refer to Table 8.1 for the meaning of the labels.

8.2.2 Environmental parameters

Similar to the simulation, the algorithm was also evaluated in the environmental conditions listed in Table 8.2. Each set of experiment was carried out five times.

Figure 8.6 shows the results of the experiments in different environmental conditions. In most of the experiments, there is a larger error on X compared to Y, similar to the simulation results. In these experiments, the amount of error is higher compared to the simulation, mainly because the maximum number of iterations was arbitrarily fixed to 50, but also because of different irregularities that happen in physical experiments. In all cases, the mean error on X is less than 1.2 m and on Y less than 0.1 m. Therefore, we can conclude that the algorithm shows robustness to the wind speed and source release rate, in the tested configurations. Further evaluations in more extreme conditions would be necessary to claim that the algorithm is robust to any environmental condition.

9 Conclusion

We successfully developed a STE method and evaluated it in simulation and in a wind tunnel with different algorithmic and environmental configurations, namely wind speed and source release rate.

One of the advantages of this algorithm is its capability to cover the three sub-tasks of the GSL problem. The enhanced navigation method that we proposed allowed for a smart search for the plume acquisition part. The amount of uncertainty associated with the estimation ensured the source declaration sub-task as well.

Thanks to the MCMC method, the high computational cost related to probabilistic algorithms was remarkably reduced and the algorithm could be run entirely on a Khepera IV robot.

Part IV STE Algorithms for Complex Scenarios

10 Introduction

Despite the growing interest in the field of GSL, application fields are often kept simplistic in the literature, due to the high complexity of the problem. However, in order to progress toward real-world deployments, algorithms must be capable of coping with more challenging environments. For an algorithm to be applicable in realistic environments, it needs to be adapted to more challenging and complicated setups.

Although 3D GSL methods are recently attracting more attention due to easily accessible flying vehicles, few works can be found addressing the problem of environment complexity and sensing system variation. Furthermore, a single algorithm, capable of coping with all three axes of complexity would be a major step toward a real-world scenario where any of the scenarios, or a combination of them, could occur.

To determine if the STE algorithm is capable of such adaptability, after having evaluated our method in a baseline scenario, we introduced new features and considered multiple more complex scenarios.

Three different axes of complexity for the scenarios are considered in the scope of this thesis:

- Spatial dimensions: performing STE in a 3D search space.
- Sensing system: deploying STE on a homogeneous multi-robot system.
- Environment: adapting the STE method to operate in an unknown and possibly cluttered environment.

For each scenario, after making the appropriate adaptations, the algorithm is thoroughly evaluated in systematic experiments. The detail of the adaptations and assessment results are presented in the remaining chapters of this part. Differently from Part III and V, since different challenges are addressed in each chapter of this part, separate conclusive sections are presented at the end of each chapter, instead of a general one at the end of this part.

11 STE for 3D Search

11.1 Introduction

Despite an important number of research on GSL, and even though the gas plume distribution is inherently a 3D phenomenon, the large majority of the algorithms have been developed for a 2D space. Additionally, most of the few works considering the third dimension, have been carried out in highly simplified environments.

In a first attempt, Ishida et al. [74] developed a sensing probe for three-dimensional gas source localization, using an ultrasonic anemometer and six gas sensors. The airflow velocity and the gas concentration gradient was measured and then combined, in order to obtain a three-dimensional vector. Bio-inspired algorithms have also been implemented in 3D in [75] and [76] where the methods of dung beetles and moths are used respectively. A 3D formation-based system is presented in [32] as a combination of ground and an emulated aerial robot.

In most of the 3D works, a simplified version of an aerial vehicle is used. The main reason is because the propellers of drones might disturb the plume while moving inside it.

In this thesis, in the interest of showing that STE algorithms are applicable to a 3D space, we have evaluated our algorithm in simulation under different environmental conditions.

11.2 Adaptations

The main difference induced by the extra dimension is due to the plume model. The Z component of the source location is taken into account as well as the parameter σ_z in the plume model, presented in Equation 7.1. However, since the plume is assumed isotropic in Y and Z directions, hence $\sigma_y = \sigma_z$, the only extra parameter to estimate is the Z component of the source location z_s.

Therefore, in the estimation part, the posterior becomes six-dimensional and in the navigation

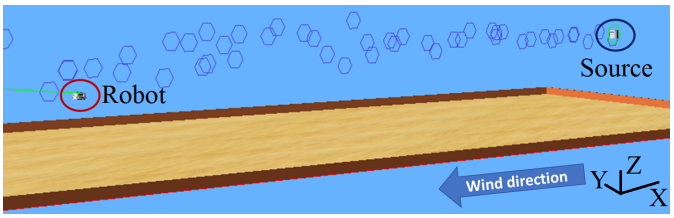


Figure 11.1 – Simulation environment in Webots, with the source upwind, the robot downwind, and the gas patches represented with blue hexagons.

part, the movement vectors become three-dimensional.

11.3 Performance Evaluation

This scenario, being a slight modification from the baseline environment is only validated in simulation. We carried out systematic experiments in simulation to study the impact of environmental conditions, i.e. the wind speed and the release rate. In the present section, we explain the experimental procedure and the outcome.

11.3.1 Simulation experiments

In this particular setup, by disabling the gravity in Webots, we were able to use our wheeled robot as an aerial vehicle to assess the algorithm in 3D, illustrated in Figure 11.1. The simulated area was designed to be 2 m high. To make the performance evaluation as fair as possible, we randomly set the initial position of the robot, as well as the position of the source on the Y- and Z-axis. On the X-axis, the source remains on extreme upwind direction, with 1 m distance from the wall.

Figure 11.2 shows a sample trajectory of the robot. In this example, similarly to the 2D setup, at the beginning of the experiment, large steps were taken to explore the environment. When the uncertainty gradually decreased, the belief on the source location approached that of the real one and, as a result, the steps became smaller around the area of the source position. Finally, for the uncertainty to become negligible, the robot sampled several points around the source area before stopping the algorithm.

Environmental parameters

We studied the performance of the method in different environmental conditions with the algorithmic parameters deduced from the 2D environment. For the sake of consistency, the environmental parameters are chosen to be the same as in the 2D setup, presented in Table 8.2.

The results of these evaluations are presented in Figure 11.3, where experiments were repeated

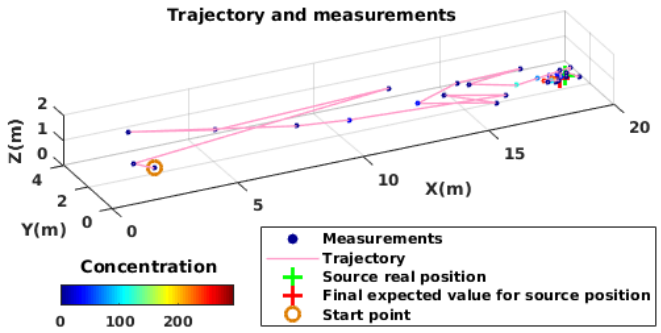


Figure 11.2 – Trajectory example of simulation in 3D.

ten times for each setup. Apart from a few outliers, the error is mostly very low on the source position and the number of iterations seems to be consistent in all environmental conditions. The error is around 10 or 20 cm on the Z-axis, which is a good performance for a 2 m range ($\leq$ 10% error). On Y- and X-axis, the error is similar to the evaluation in the 2D setup, shown in Figure 8.3.

Moreover, as in the previous experiments, we can see that the general performance is not affected by environmental conditions. However, unlike the 2D setup, a few outliers can be observed with very high estimation error. This might be caused by the higher complexity level of the search due to the third dimension.

11.4 Conclusion

The presented results show that the STE algorithm is inherently able to operate also in a 3D space. Experiments in a real 3D environment with a flying vehicle would be necessary to show whether the algorithm would be still efficient when the airflow produced by the propellers of the flying vehicle disturbs the gas plume. We expect that the measurement error of the likelihood should be set to a higher value to compensate this effect. Moreover, if the effect is so intense that the plume shape is also affected by this disturbance, then the model error should be set accordingly as well.

From a system point of view, having a method that is versatile in terms of covered dimensions is a major convenience for heterogeneous systems where both ground and flying vehicles can cooperate through the same algorithm.

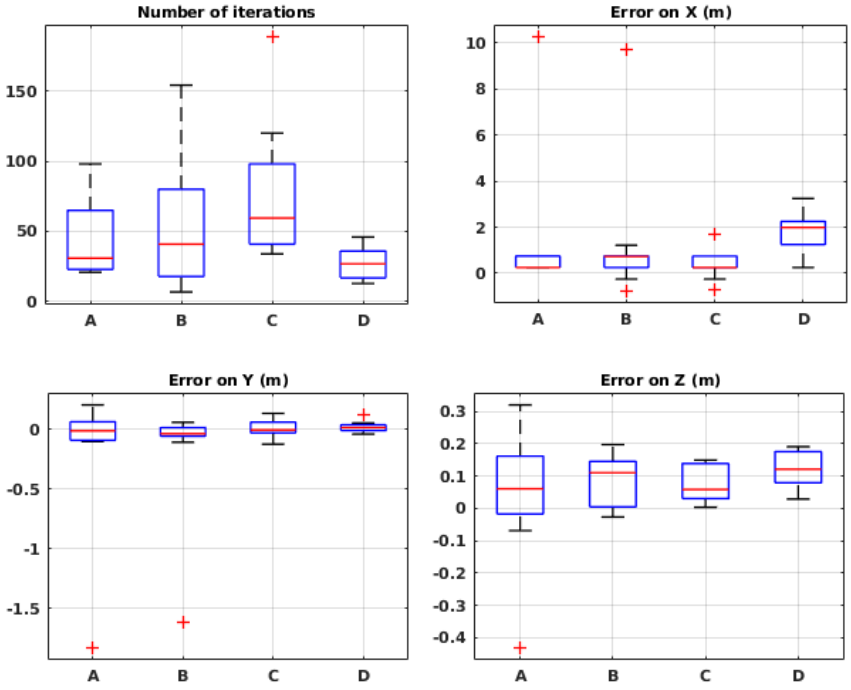


Figure 11.3 – Evaluation results in simulation for different environmental conditions in the global framework. See Table 8.1 for the meaning of the labels.

12 STE for Unknown Environments

12.1 Introduction

Most of GSL applications target time-critical and punctual situations, where a predefined map of the environment might not be available. Moreover, in case of an indoor environment, as the GNSS-based techniques will not function properly and it is unlikely that the place is equipped with indoor absolute positioning systems, the robot cannot localize itself in the arena. Therefore, it seems important for such methods to be compatible with a priori unknown environments.

In the previously presented implementation of the STE algorithm, for ease of experimentation, the robot was assumed to have access to the map of the environment as well as its absolute position in the map using a global positioning system. In this chapter, we present an adaptation of the algorithm that makes it possible to function without a global map and an absolute positioning system, which we call a local framework, as opposed to a global one.

In the local framework, the robot uses odometry to localize itself, with the origin being at the starting point in the arena. To cope with lack of a global map, the robot creates a fixed-size local map (e.g., $2 \times 2\,\mathrm{m}^2$ in this work) aligned with the wind direction around itself and performs the estimation and navigation on that local map using odometry. Once enough information is acquired on the local map, it then moves to another place.

Even though the two frameworks seem very different in terms of localization method, we applied the same concept on both of them and there are only a few minor differences in the implementation, mainly in the navigation part, which we discuss in detail in the following section.

12.2 Adaptations

The navigation method presented in Section 7.3 for the global framework is also valid for the local one. However, in the case of the local framework, we need to introduce additional

algorithmic features in order to cope with the lack of global localization and environment map. A summary of all of them can be found in Algorithm 1.

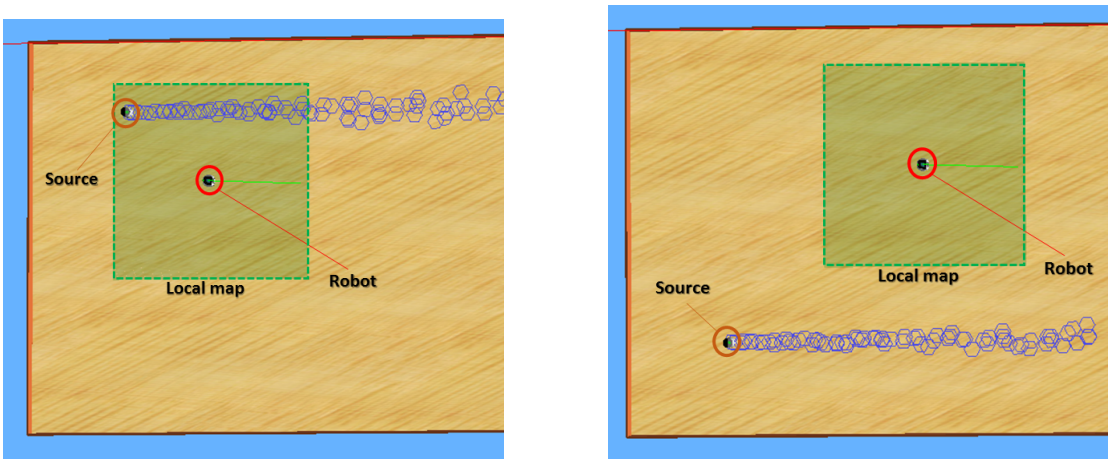
As mentioned above, since the robot does not have the map of the environment, it has to generate a local one as a base for its estimation and navigation. The same procedure as the global framework is then applied to the local map with the exception that it is done at a smaller scale and in different steps. More concretely, the algorithm is run on one local map, and then the map is moved to another place where the robot restarts from the beginning. The cycle continues until the source is localized inside the map and thus the entropy of the posterior probability distribution drops below a fixed threshold.

Unlike in the global framework, where the shape of the environment is known a priori by the robot, in the local framework, while navigating towards a target point to sample, the robot might encounter an obstacle, for instance a wall. In this work, for the sake of simplicity, when the robot encounters an obstacle on its way, it simply stops after having carried out an obstacle avoidance maneuver (in our case, we used a simple Braitenberg algorithm [77]), and then sets the measurement of the target point to 0. In this way, that point and its neighboring points are not considered as sampling point candidates. Then, the cycle continues normally.

Another process that is special to the local framework is the way the robot decides to move from one local map to the successive one. Conceptually, this should happen when there is enough information for the robot to move on. However, since the error on odometry accumulates with every step, we wish that the robot resets all position-based information (i.e., self-localization and samples positions) in the new local map to reduce the influence of odometry error on the performance. Therefore, moving to a new map means erasing all the previously acquired information. Hence, spending too much time in one map would be a waste of time. Spending too little time, on the other hand, although it reduces the number of iterations, could be misleading. As a result, the maximum number of samples in the local map is one important factor that we will study in this chapter.

Moreover, sampling the same number of points in all the local maps does not appear to be optimal. Depending on where the local map is located with respect to the source location and the plume in general, the robot might need to sample more, or fewer points. The case that illustrates this situation is when the source is located inside the local map. In this situation, we would want the robot to continue sampling in the same map until the end condition is satisfied, rather than move to another map (which will be most likely again around the source) and start over. The opposite case is when the local map is completely outside the plume and the robot would sense no concentration at any point of the map. In this situation, we would like the robot to move on to the next map as soon as possible, where it might have a better chance of obtaining relevant information. Figure 12.1 illustrates the two mentioned examples.

Based on the arguments above, we adapted the number of samples in each map to the amount of information that we obtain through sampling. More specifically, we define an initial sample budget B_0 that represents the number of samples that the robot would make if no information



(a) (b)

Figure 12.1 – Position of the local map with respect to the source and the plume: (a) source inside the local map of the robot and (b) local map outside the plume.

was available. The budget is then reduced every time one sample is made. However, this reduction is not the same at every iteration and depends on how valuable the newly obtained sample was. If the sample does not bring any information, i.e., the entropy stays the same as the previous iterations or even increases, the budget is reduced by 1. If the entropy drops, on the other hand, the budget would be reduced by less, depending on the amount of obtained information. Equation (12.1) shows the mathematical definition of the budget reduction. Note that $H_{previous}$ can never be equal to 0 because, as explained in Section 7.4, as soon as the entropy drops below a certain positive threshold, the algorithm stops. The optimal value of the initial sample budget B_0 is studied further in this work.

$$B_t = \begin{cases} B_0 & \text{if } t = 0 \\ B_{t-1} - 1 & \text{if } \frac{H_{current}}{H_{previous}} \geq 1 \\ B_{t-1} - \frac{H_{current}}{H_{previous}} & \text{otherwise} \end{cases} \tag{12.1}$$

Once we know when to move to a new map, we need to know the best point to move to. For the reasons mentioned in Section 7.3, it appears that the maximal a posteriori value of the final posterior probability density function would be the best option. Indeed, we need to explore the environment as much as possible when we receive very little information and we want the robot to get closer to the source step-by-step when it has enough information. Using relative entropy here would not be useful since, when we move to the new map, the previous points would already be forgotten. Therefore, here pure exploitation seems to be preferable. Thus, when the robot decides to move to a new map, it goes to the point indicated by the maximal a posteriori value of the posterior probability density function, which becomes the center of the new map. From that point, all the procedure starts over. Figure 12.2 shows a graphical

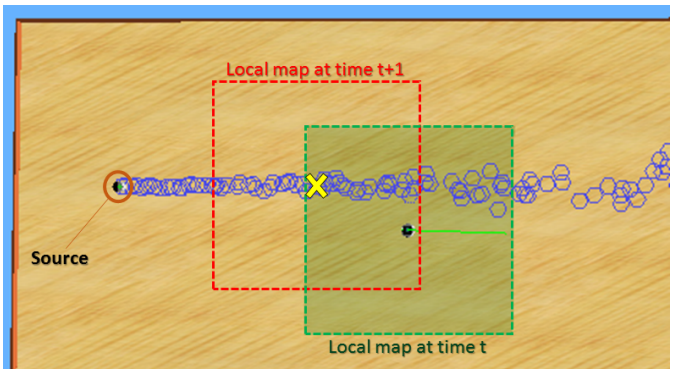


Figure 12.2 – At time t, the local map is located at the position of the green square. The yellow cross shows the maximum a posteriori value of the posterior probability density function, which is the point that is going to become the center of the local map at time $t+1$. When the robot decides to move the map, it travels to the yellow cross point, and it restarts the procedure in the local map indicated by the red square.

representation of the move from one local map to another.

In the same line of exploitation, when no concentration is sensed in one map, there is still one source of information that can be exploited, which is the wind direction. Therefore, to increase the exploitation, our method moves the map crosswind when the sum of all sensed concentrations is equal to 0. Whether to move left or right along the crosswind direction is chosen randomly in a uniform way, and can be changed at each iteration.

Algorithm 1: Local framework: when and how to change the position of the local map

```
initialization;
while Entropy > entropy_threshold do
    if Sample_Budget > 0 then
        if obstacle encountered then
            Stop the robot;
            Measurement[goal_position] = 0;
        else
            Make measurement;
        end
        Estimation;
        Update goal_position;
        Move to goal_position;
        Update Sample_Budget;
    else
        if sum(measurements) = 0 then
            new_map_position = crosswind;
        else
            new_map_position = maximum a posteriori;
        end
        Reset Sample_Budget;
    end
end
```

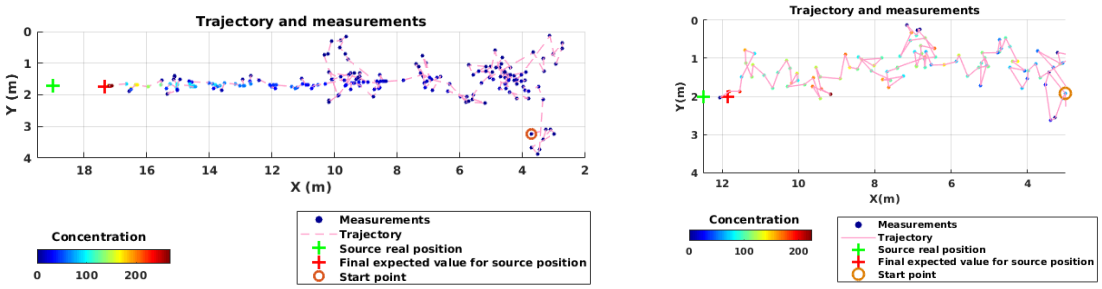
12.3 Performance Evaluation

To assess the performance of the algorithm in the local framework, we evaluated it in the very same setups used for the global framework.The underlying algorithm and experimental environments being identical in both frameworks, we used the chosen values of the parameters that we previously studied, namely the exploration–exploitation ratio α and the likelihood's model and measurement errors σ_M and σ_D.

12.3.1 Simulation experiments

For the local framework evaluation, we used the same simulation environment in Webots as in the global framework described in Section 8.1. Due to the characteristics of the framework, the robot localized itself using odometry, and its navigation and estimation field was limited to the local map, shown by the green square in Figure 12.1. The size of the local map 2×2 m^2 was chosen in order to cover a small portion of our 20×4 m^2 arena, and, at the same time, to leave enough space for the robots to explore.

Figure 12.3a shows a sample trajectory of the robot in simulation. The samples are shown



(a) (b)

Figure 12.3 – Two examples of the robot's trajectory in the local framework: (**a**) simulation, and (**b**) wind tunnel. The initial position of the robot is shown with an orange circle; the actual source position (green cross) as well as its estimated location (red cross) are also indicated. On each sampled point, the measured concentration is indicated by the color of the circle, which corresponds to the averaged value of the measurements in a 5-s time window. The measured concentrations are relative values given by the sensor in mV.

along with the robot's trajectory. As expected, the robot took small steps and changed its course usually towards the source position. At the end, the robot stopped very close to the source where the algorithm reached its end.

Algorithmic parameters

As mentioned above, three algorithmic parameters were studied in the global framework, and we simply used their chosen values for the local framework. However, in the local framework, we introduced a new algorithmic parameter, namely the initial sample budget value B_0, that we still need to study.

Figure 12.4 shows the performance of the method with different values of B_0, each repeated ten times. As expected, the number of iterations systematically increases with higher values of the initial budget. The source position estimation error, on the other hand, shows different results: with very low values of sample budget, the error is very high, since the algorithm does not obtain enough information to guide the robot in an efficient way. With values between 7 and 9, the error drops to acceptable values (1 to 2 m of error on the X-axis). For larger values (e.g., between 25 and 50), however, we notice a slight increase in the error value on the X-axis. We believe that this error is due to the accumulated odometry error after so many steps. For the rest of the evaluation, we set the initial budget value to 7, which seems to be a close to optimal value, according to the presented results.

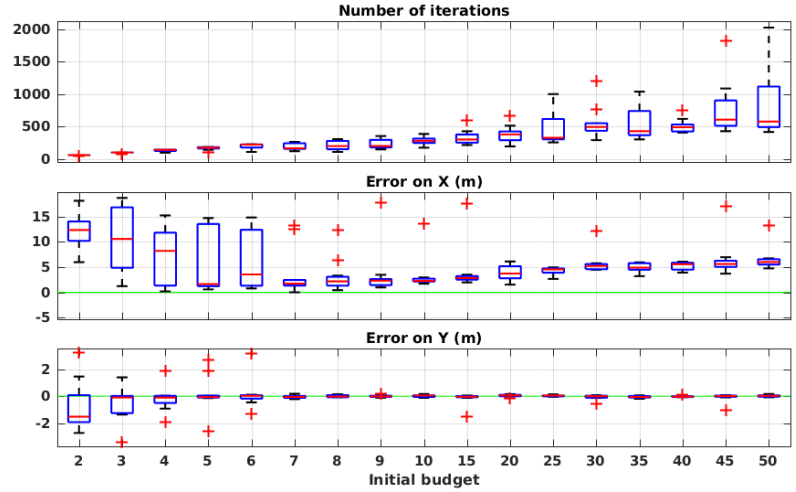


Figure 12.4 – Evaluation results in simulation with different initial values of the sample budget.

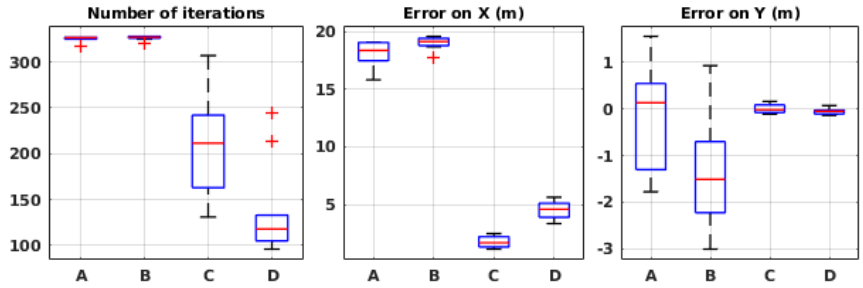


Figure 12.5 – Evaluation results in simulation for different environmental conditions in the local framework. See Table 8.1 for the meaning of the labels.

Environmental parameters

To assess the robustness of the method to environmental conditions, we evaluated the performance of our method in the local framework in the same conditions as in the global framework, which are listed in Table 8.2.

Figure 12.5 presents the performance of the system in ten runs for each environmental condition. While showing satisfactory results in some situations, the algorithm fails in others. It appears that, in high wind speed (conditions C and D), the error is in an acceptable range, but, in low wind speeds, the algorithm fails. We believe that it is due to the limited estimation field. Indeed, when the robot is far from the source, in some situations, the posterior probability distribution created on the local map predicts in which orientation the source is located and therefore guides the robot towards it. In low wind speed, on the other hand, this does not happen because the plume is much wider and thus it guides the robot less accurately, and, since the robot does not keep the outcome of the previous local maps, it cannot recover after being misguided.

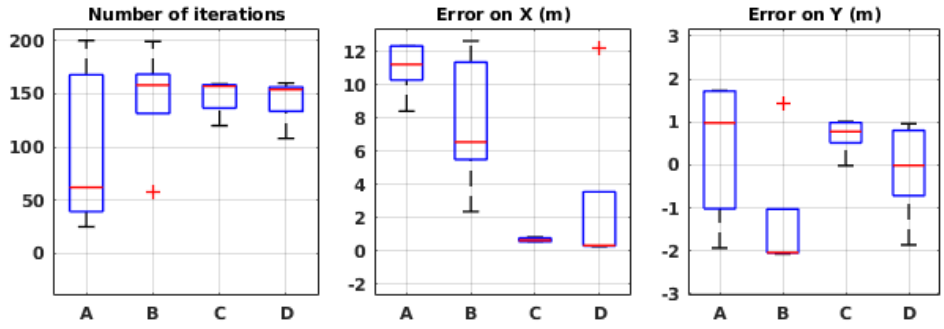


Figure 12.6 – Evaluation results in wind tunnel experiments for different environmental conditions in the local framework. See Table 8.1 for the meaning of the labels.

12.3.2 Physical experiments

The algorithm was also tested in the local framework in a wind tunnel. The same setup and environmental conditions were used as in the global framework. The results, as shown in Figure 12.6, seem consistent with the ones from the simulation in Figure 12.5. Indeed, in high wind speed, the mean error is 0.25 m on the X-axis and less than 1 m on the Y-axis. However, again, in low wind speeds, the algorithm seems to be misguided and fails. With the confirmation of physical experiments, we know that the algorithm is not robust to different environmental conditions in its current implementation.

In Figure 12.3a, a sample trajectory of the robot in a physical experiment is shown. Similar to the sample trajectory from a simulation experiment shown in Figure 12.3b, the robot takes short steps and finishes the task very close to the source.

12.4 Discussion

The algorithm shows robustness to the tested environmental conditions in the global framework, but, in the local framework, it only works in high wind speed. We believe that it is because, in the global framework, the source is certainly located in the map, where the estimation takes place. In the local framework, the local map does not initially enclose the source, but, using the estimated posterior distribution, it supposedly guides the robot towards the source.

One advantage of the local framework that we did not address is the low travelled distance. In Figure 12.7, the total travelled distances in both frameworks are shown in simulation in condition C. It appears that, in the local framework, on average, the robot moves three times less than in the global framework. The limited navigation strategy in the local framework, on the one hand, and the large exploration steps in the global framework, on the other hand, are the causes of the difference between the two frameworks.

In [78], we proposed a 3D gas source localization algorithm based on Infotaxis, leveraging a

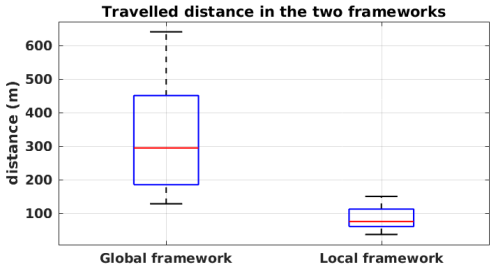


Figure 12.7 – Travelled distance in simulation for the environmental condition C in both frameworks.

similar global framework to that used for the STE algorithm in this thesis. They both rely on a gas plume model, but, in the Infotaxis algorithm, all model parameters but those defining the source lcoation are assumed to be known before the experiment. This assumption, while being applicable for a known environment, limits the usage of the method in real world scenarios where there is usually no a priori information about the plume. The STE algorithm, on the other hand, does not use a priori information on the parameters of the model and estimate the values based on the acquired data. The only predefined values are the limits on the range of each parameter. Thus, by choosing realistic yet meaningful ranges, the algorithm can be run in different configurations.

It is worth mentioning that a few assumptions are included in the STE algorithm, as explained in Section 7.2. However, they are different from a priori information which are meant to simplify the evaluation. In this work, by making assumptions, we did not seek to simplify the evaluation of the algorithm, but only the algorithm itself and its computation. For example, assuming that the plume is isotropic (i.e., $\sigma_y = \sigma_z$) was meant to decrease the number of parameters to estimate, hence, to simplify the computation. Such simplifications might not match with the real setup and thus the algorithm would work less efficiently. However, since the algorithm showed success with these limitations, we can expect that it would work more efficiently without them. The same reasoning is valid for the choice of the model, and the assumption based on which it is driven.

Another interesting difference between this work and our previous one [78] is that, in that work, the estimation was done on the entire map of the environment (globally) while the navigation was done in small steps, i.e., more similar to our local framework. The results of that work showed lower performance on low wind speed as well compared to high wind speeds. However, the mean of error on source localization was lower in [78] than in the present paper. Therefore, a combination of local navigation and global estimation seems to be more suitable for this type of algorithms.

12.5 Conclusion

We successfully adapted the proposed STE method on the local framework in simulation and in physical reality. The impact of different algorithmic parameters were studied and the robustness of the method to different environmental configurations, namely wind speed and source release rate, was evaluated.

The successful evaluation of this method in the local framework, without any predefined environment map and any global positioning system, shows the high potential of this algorithm for unknown environments. However, its downside is the lack of robustness to different environmental conditions, which could be studied in future work.

13 STE for Multi-Robot Systems

13.1 Introduction

Thanks to the flexibility of the STE algorithm, it can also be used with a distributed sensing system. There could be several advantages in using a distributed sensing system for such task, for instance, reducing the necessary time to achieve the task, making use of diverse resources, or distributing sub-tasks among assets throughout the procedure. However, for a distributed sensing system to be advantageous in comparison to a single sensing asset, it must operate with an appropriate degree of coordination. Therefore it is important to understand how much the coordination helps the performance of the system.

In related works, such as [55] and [79], STE-related algorithms are used with a distributed system to achieve source localization. However, in [79] the algorithm is only evaluated in simulation using bodiless mobile sensors as well as an field data base. Additionally, the mobile sensors are supposed to be working in a synchronous fashion, which could be problematic in a real environment. In [55], the algorithm is evaluated using real robots and a humid-air plume, but the motion coordination was fully centralized and the advantage of the coordination was not sufficiently investigated and thoroughly benchmarked with a single-robot system.

In this part, we present three levels of coordination strategies coupled with our STE method deployed on a homogeneous multi-robot system. The algorithm is run in a distributed fashion, therefore each robot runs the algorithm individually, while receiving information from other robots according to the coordination strategy. Each strategy is evaluated in simulation and physical experiments using up to three robots.

Additionally, in this part, the previously used navigation strategy is enhanced to reduce the traveled distance even for a single robot. The new navigation strategy is evaluated separately along with its key parameters.

In the remaining of this chapter, we will present the coordination strategies, the evaluation metrics, the evaluation process and we present the obtained results. Lastly, in the conclusion,

we will discuss the outcome and present the outlook for this work.

13.2 Enhanced Navigation Method

In previous parts, we defined the movement vector as a weighted sum of two vectors: the one leading towards the target point that provides more information, given by Kullback–Leibler divergence calculation, $\vec{V}_{KLD}$ and the vector that leads to the maximum a posteriori value of the source position $\vec{V}_{source}$.

$$\vec{V} = \alpha \vec{V}_{KLD} + (1-\alpha)\vec{V}_{source} \tag{13.1}$$

In this part however, with the intention of reducing the travelled distance by the robot, we also take into account the distance of candidate points from the current robot's location. In fact, when the amount of uncertainty is still high because of lack of information, due to the stochastic nature of the MCMC algorithm, the maximum a posteriori value of the source position could change very quickly. This, indeed, promotes exploration, but makes the robot travel very far while a closer point could have a similar value.

Therefore, instead of leading the robot towards the maximum a posteriori value of the source position, we create a combined map of two density functions defined on every grid cell of the environment map: $P_M(x_s, y_s)$, defined in Eq. (13.2), is the marginal probability density function for the source position, and $D_I(x, y)$ represents the inverse of the distance of every point to the current position of the robot, as defined in Eq (13.3).

$$P_M(x_s, y_s) = \sum_Q \sum_{\sigma_a} \sum_{\sigma_b} P(Q, x_s, y_s, \sigma_a, \sigma_b) \tag{13.2}$$

where $P(Q, x_s, y_s, \sigma_a, \sigma_b)$ is the posterior density function for all the source parameters.

$$D_I(x, y) = 1/\sqrt{(x-x_r)^2 + (y-y_r)^2} \tag{13.3}$$

where (x_r, y_r) is the current position of the robot. D_I is normalized in such a way that the sum of all the values will be equal to 1. D_I is set to 0 for the cell that the robot is currently positioned on, in order to avoid division by 0.

In order to combine the two density functions we use a weighted sum defined in Eq.13.4 where $\beta \in [0, 1]$.

$$P_{combined}(x, y) = \beta P_M(x, y) + (1-\beta) D_I(x, y) \tag{13.4}$$

The combined density function contains higher values for points that have a high potential of

Table 13.1 – Summary of Coordination Strategies

Strategy	Nb of robots	Coordination
Individualist	1, 2, 3	Source Declaration
Cooperative	2, 3	Source Declaration Sample Sharing
Collaborative	2, 3	Source Declaration Sample Sharing Movement Coordination

containing the source and that are closer to the robot. Therefore, if two points have the same probability of containing the source according to the posterior, $P_{combined}$ attributes a higher value to the one that is close to the current position of the robot. Once the maximum point of $P_{combined}$ is found, $\vec{V}_{source}$ of eq. (13.1) will point to it.

The value of coefficient β is determined experimentally and will be discussed in Section 13.4.

13.3 Coordination Strategies

There are multiple stages on which the robots can join forces in this framework: sharing samples, coordinating navigation, and source declaration. For a better understanding of the benefit of each coordination technique, we suggest different strategies with different levels of coordination. Then each strategy is evaluated with up to three robots. The characteristics of the strategies are explained in the following and a short summary is given in Table 13.1.

13.3.1 Individualist strategy

In this scenario, every robot performs the algorithm independently, i.e. they do not share any information with the other teammates. However, the source declaration of one robot is enough for the entire team to end the search. Indeed, although this strategy involves a minimalistic level of coordination, the robots still form a team, and if one of the teammates reaches the required level of certainty to declare the source, the other ones do not need to continue the search.

13.3.2 Cooperative strategy

Cooperation means sharing information between teammates in support of each other's goals. Therefore, in this scenario, the robots share their acquired samples with the other robots. Since we suppose a reliable communication among the robots, they are all supposed to have the same data to calculate the posterior. However, since the estimation is done stochastically using the MCMC algortihm, the outcomes could be slightly different from one robot to another. In this scenario, no coordination is considered for navigation but the source declaration is done as a team like in the individualist scenario.

13.3.3 Collaborative strategy

Collaboration is working together in support of a shared goal. In the collaborative scenario, the robots not only share their samples with each other as in the cooperative strategy, but also coordinate their movements together.

The movement coordination can be defined in different ways. Here, with the intention of saving time and resources for the robots, we chose to make use of a task allocation technique: considering that the current goal of each robot is a task, it can be achieved by any other robot, since any collected sample is shared with all the robots. Therefore, every time a robot decides a new goal position, it verifies if another way of distributing tasks between the robots would decrease the total travelled distance by the team. If so, it makes the necessary swaps and informs the involved teammates to change their goals. To achieve this, the robots need to share their current position, as well as their current goal position with each other at all times. The swap can only happen for a robot if it is moving towards its goal.

Additionally, in this scenario, since the robots have the information about their teammates' current goal positions, they will take them into account before deciding their next move. In the estimation part, they temporarily include a prediction for the outcome of the samples that the other robots are going to collect in their goal positions and in this way they reduce the risk of going to the same point. The prediction is made using the current belief on the source parameters.

Similarly to the cooperative scenario, the communication is assumed fully reliable and fast. In any case, the algorithm success does not depend on inter-robot communication. However, the advantages brought by coordination cannot be leveraged until the communication failure is resolved.

13.3.4 End of algorithm

As mentioned in Section 7.4, the algorithm stops when the uncertainty on the source parameter estimation becomes very low.

In case of a multi-robot system, when the source is declared by one robot, all the other robots stop the process as well. The entropy could be slightly different for different robots even when all the acquired data are shared because of the stochasticity of estimation caused by the MCMC algorithm. However, when one of the teammates is certain of its estimation, it is assumed to be enough for the entire team.

Additionally, there is also a timeout that forces the algorithm to stop when the estimation takes more time than expected which is represented by a pre-established total number of iterations. This timeout, while being the same for all the robots, does not necessarily happen for all of them at the same time because of the assynchronicity of the team. When one robot reaches its timeout, it does not force the others to stop as well, they can carry on hoping that one of them

reaches the certainty threshold before the timeout.

13.4 Performance Evaluation

Similarly to previous parts, we first implemented and evaluated the method in simulation, and then in the physical reality.

The metrics used for the evaluations are as follows:

- Number of iterations: it is the sum of the number of iterations of all the robots, which measures the total computational cost.
- Experiment duration: it is the time measured from the start of the experiments until the first robot declares success and ends the experiment. This metric measures the response time of the method.
- Travelled distance: it is the sum of the travelled distances by all the robots, which measures the total energy cost.
- Estimation error: it is the Euclidean distance between the real source position and the estimated one. This metric measures the accuracy of the method.

In all four metrics, lower values are better. For single-robot experiments, the experiment duration is not shown because it is proportional to the travelled distance. In strategy comparisons, the respective estimation errors are not shown because they were all similarly very low.

In the remaining of this section, we explain the experimental procedure as well as the outcome.

13.4.1 Simulation experiments

Similarly to previous setups, we first evaluate our method in simulation. The first sets of experiments in simulation are intended to tune the algorithmic parameters of the method. Once the best parameter values are deduced, the algorithm is assessed in terms of performance with different coordination strategies and different number of robots.

Algorithmic parameter

Before evaluating the algorithm in different scenarios, we studied the influence of the ratio β from Eq. (13.4). Using a single robot, we run ten experiments for different values of the parameter β. The results presented in Figure 13.1 show that the best compromise value is 0.75, since it reduces the travelled distance, but lets the robot travel far enough to find the source with a low error.

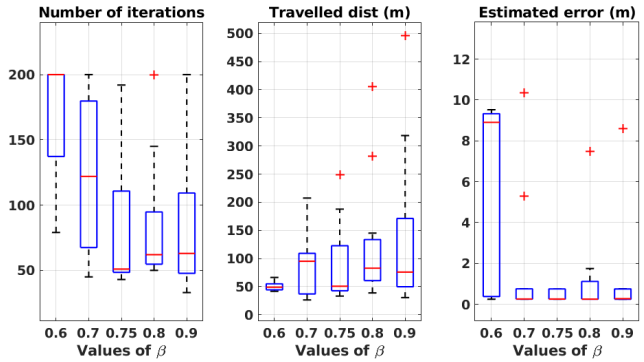


Figure 13.1 – Performance evaluation in simulation with different values for β in eq. (13.4).

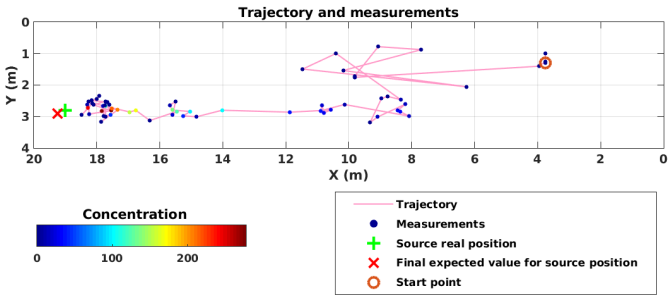


Figure 13.2 – An example of trajectory performed in simulation. The start point, the trajectory, the measurement points as well as the source position are shown on this figure. The search ended very close to the source.

Figure 13.2 shows a sample trajectory of one robot on simulation. In this example, the robot started far from the source and outside the plume. At the beginning of the experiment, large steps were taken to explore the environment to find the plume. Once the uncertainty gradually decreased, the steps became smaller. Finally, for the uncertainty to become negligible, the robot sampled several points around the source position area before stopping the algorithm.

Coordination strategy evaluation

Once the best value for the algorithmic parameter is found, we studied the performance of the algorithm using the different strategies that we defined in Section 13.3. Each coordination strategy is evaluated with up to three robots, and each experiment is repeated ten times. Figure 13.3 shows the results of this evaluation.

In the individualist strategy, the total number of iterations and the total travelled distance increase with the number of robots. The reason is that in the absence of coordination, more active robots travel more distance together compared to a single robot. The experiment duration, however, does not follow this trend. Its median value stays similar, while the variation drops with higher numbers of robots. This is due to the source declaration that is done as a team. With higher number of robots, and due to the stochasticity of the estimation process,

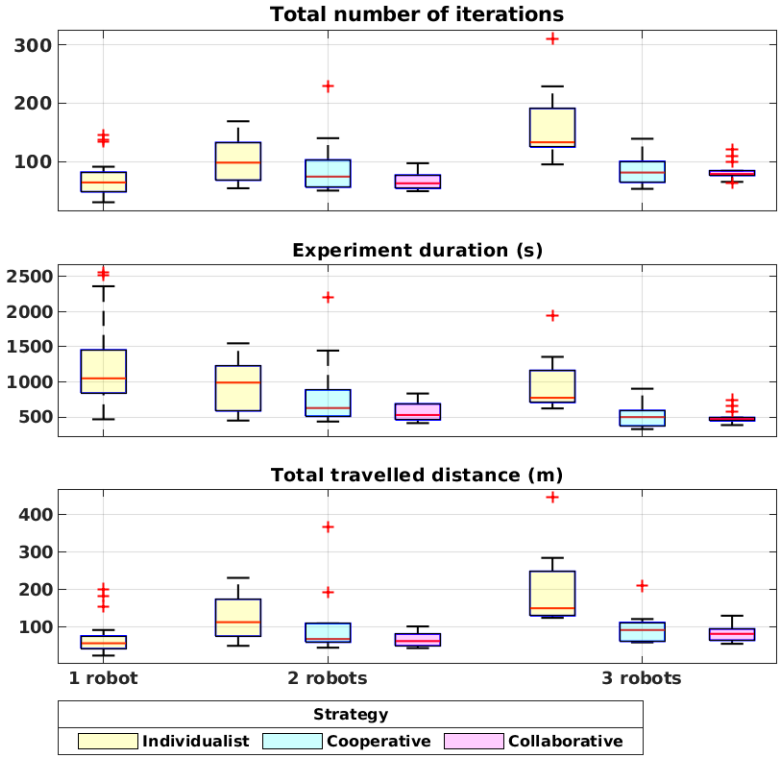


Figure 13.3 – Evaluation results of the algorithm in simulation with the strategies defined in Table 13.1.

there is a higher chance that a robot arrives at the required certainty level and declares the source position. Therefore, even with no coordination among the robots throughout the experiment, having more robots may reduce the necessary time to locate the source.

In the cooperative strategy, since the robots share the acquired data, they tend to reach the same conclusion about the estimated source position. The median of the number of iterations does not change much compared to the case of a single robot. This is due to the fact that the total number of iterations is equal to the number of acquired samples, and since all the collected samples are accessible for all robots, their number should be in the same range as the one of a single robot. For the same reason, the sum of all travelled distances follows the same trend. The experiment duration, however, decreases with higher number of robots since different points of the environment can be sampled simultaneously.

The collaborative strategy follows the same trends as the cooperative strategy in all metrics, in comparison with the individualist strategy. A slight reduction in travelled distance and experiment duration can be seen when compared to the cooperative strategy which is due to the task allocation technique. Based on intuition, we believe that if the distance component was not introduced in the navigation, the steps would be longer (see results in Chapter 8) and therefore, we could see a higher influence of the collaborative strategy when compared to the cooperative one.

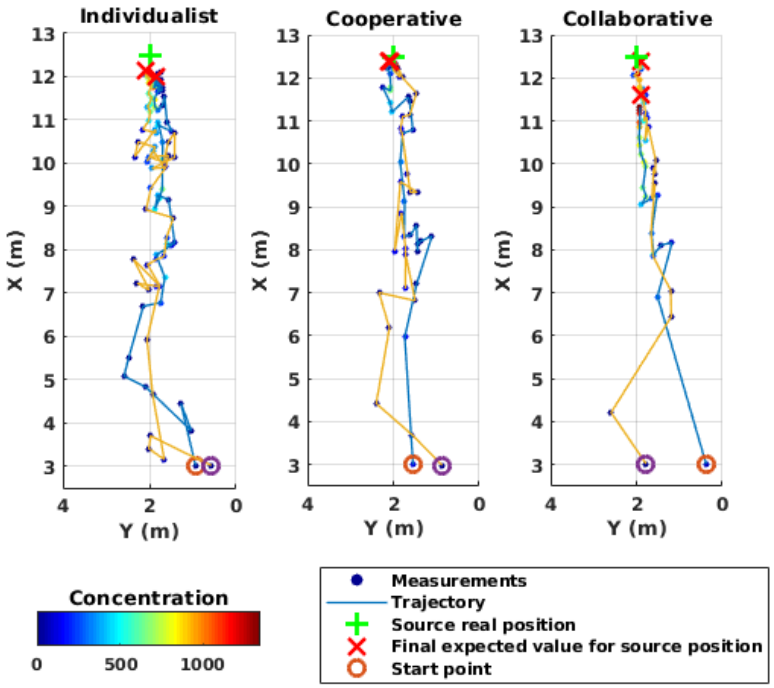


Figure 13.4 – An example of trajectories performed in the physical experiments with two robots. The start point, the trajectory, the measurement points as well as the source position are visible in this figure. The search ended very close to the source.

13.4.2 Wind tunnel experiments

Figure 13.4 shows one sample trajectory for each strategy with two robots in the wind tunnel. Similarly to the simulation experiments, in all the strategies, first, the robots starts to explore the environment by taking large steps. Once they find the plume and have less uncertainty about the source parameters, the steps become smaller and the overall movement is oriented towards the source. The final expected source position is usually very close to the real one. It is worth noting that the more coordination is involved in the experiment, the smoother the trajectories become. In the individualist scenario, both robots are alone in performing the algorithm, therefore each of them needs to collect enough samples individually. In the cooperative scenario, the data are shared, therefore each robot needs to collect less samples and as a result, they progress faster toward the source. In the collaborative scenario, the trajectories rarely cross each other, which makes the robots to have less collision risk and less travelled distance.

Figure 13.5 shows the performance of the algorithm in different strategies with different number of robots. Each set of experiment was repeated five times. Similarly to the simulation results, the experiment duration decreases with higher levels of coordination and higher number of robots. Travelled distance and number of iterations substantially increase in the individualist scenario, but by very little in the cooperative and collaborative scenarios.

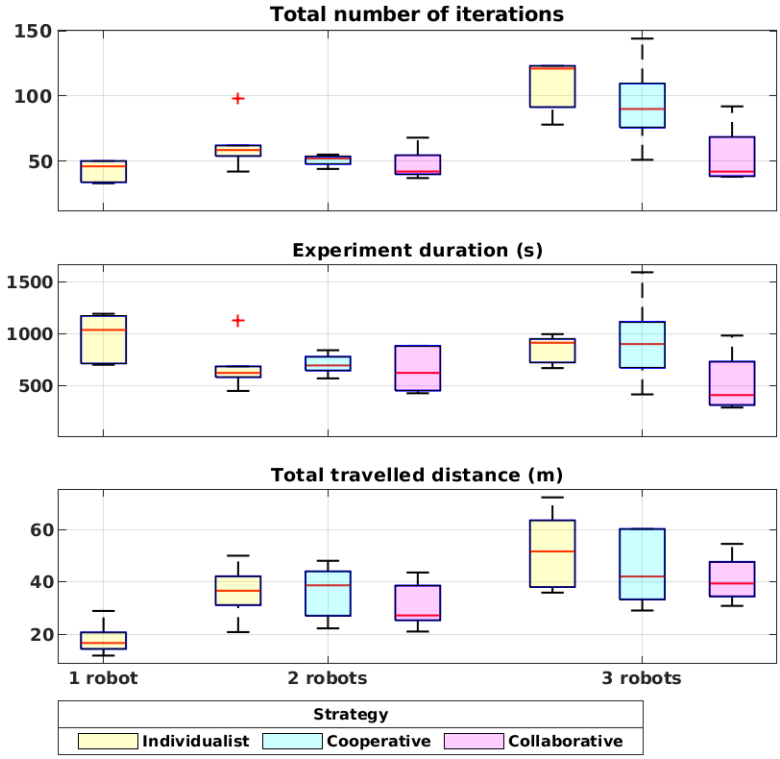


Figure 13.5 – Evaluation results of the algorithm in the physical experiments with the scenarios defined in Table 13.1.

13.5 Conclusion

We successfully evaluated a distributed implementation of our STE algorithm using a multi-robot system. The system was evaluated both in simulation and physical reality. The enhanced navigation strategy allowed for a smoother trajectory compared to our previous work [80]. Different coordination strategies were proposed in order to verify the advantages and constraints of using a distributed sensing system and different coordination strategies.

14 STE for Cluttered Environments

14.1 Introduction

Many of GSL applications take place indoors where the airflow is not characterized by a uniform laminar wind. Irregular shaped rooms and the presence of obstacles determine the airflow, and hence the gas plume, around them. Successful GSL methods, in cluttered environments, are either reactive to local changes and follow the cues to the source, such as bio-inspired algorithms [18], or if they need to localize the source in a global perspective, such as the STE algorithm, the estimation needs to be based on an adapted plume model.

In our proposed method, as well as in most STE implementations in the literature, the leveraged model, while being successful in uncluttered environment, does not take into account the characteristics of the environment and its impact on the plume. In fact, in the Pseudo-Gaussian plume model, the wind is assumed to be laminar, unidirectional and non-zero, which makes it unsuitable to cluttered environments.

Many mathematical plume models exist for gas dispersion in a time-dependent fashion, such as the filament-based model [2] or the SCIPUFF model [81], which can faithfully simulate the dispersion of a known gas release in any environment, by following gas particles or puffs from the source, at the release time, through the air, with respect to the local wind velocity.

However, for our STE algorithm, we need a steady-state plume model that takes into account the characteristics of the environment and the source, to return the predicted concentration value at the desired location. To the best of our knowledge, such model does not exist in the literature. Therefore, the first objective in this part of the thesis is to design a steady-state plume model adapted to cluttered environments.

Since such model would be very complex to design in analytical terms, we will leverage simulated data and machine learning techniques to generate a surrogate plume model. Once the plume model is produced and evaluated, we integrate it in our STE framework and assess its usability.

The remaining of this chapter will describe the design and evaluation details of the generated plume model as well as its evaluation.

14.2 Data-driven plume model

A Data-Driven Plume Model (DDPM) is a program that predicts the gas concentration based on previously seen and learned samples. To train such a surrogate model, we need a diverse dataset representing various factors that influence the shape or the scale of a gas plume. Additionally, a learning architecture is necessary to achieve the prediction. In this section, we first explain how the dataset was generated, then we present our learning framework, and finally, after evaluating the produced model, we present and discuss the results.

14.2.1 Datasets

The first ingredient that we need to train a data-driven model is abundant and reliable data. We have chosen to use simulation to gather data because it is faster, more repeatable and more efficient compared to physically gathering samples. Moreover, we have leveraged our simulation environment which is calibrated with our experimental setup to do so, therefore, the simulation data is expected to be correlative to the real-world samples.

For the training dataset to include a large variety of setups, we generated samples randomly. Additionally, we have selected nine test maps, with various complexity levels, to evaluate the performance of the model on intuitive maps. In this section, we will explain in detail how the data were generated and used.

Training dataset

The ideal training dataset for this machine learning task would be a set of gas concentration maps created in various environments, with or without obstacles, in which the gas source is placed in different locations. Therefore, to generate such dataset, we first need to create environments with different obstacle layouts. Then the maps should be fed into OpenFOAM to create the wind map corresponding to each environment. Finally, the wind map should be loaded in Webots to generate a plume and, hence, a gas concentration map, with the gas source placed randomly in the environment.

In order to create cluttered environments, we need to randomly create obstacles in a $10\times 4\,\mathrm{m}^2$ environment. For simplification purposes, we only consider rectangle-shaped obstacles, which is not far from realistic indoor environments. However, the shape and the position of the obstacles are chosen randomly. In each map, there could be one rectangle, two different and separated rectangles, as well as L-shapes and U-shapes. A few examples of these randomly generated maps are shown in Figure 14.1.

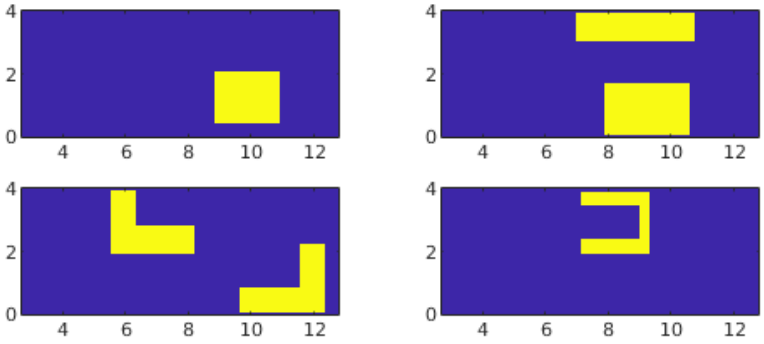


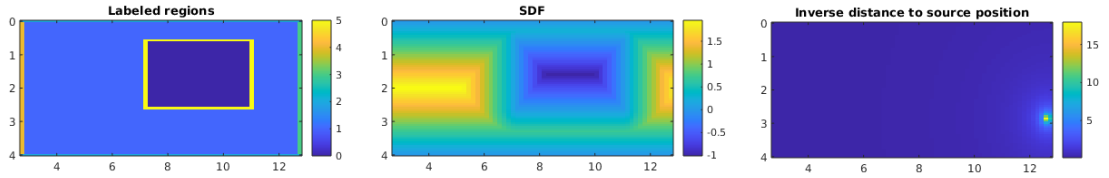
Figure 14.1 – Examples of randomly generated environments with obstacles.

180 maps were created and fed into OpenFOAM to generate wind maps. To have comparable environments, the boundary conditions, including the location of wind inlet and outlet, were identical in all maps. We chose a constant velocity of 1 m/s on the inlet (left wall), a zero-gradient condition on the outlet (right wall), and no-slip boundary condition on the top and bottom walls as well as on obstacle edges. Therefore, we obtain one wind map for each environment.

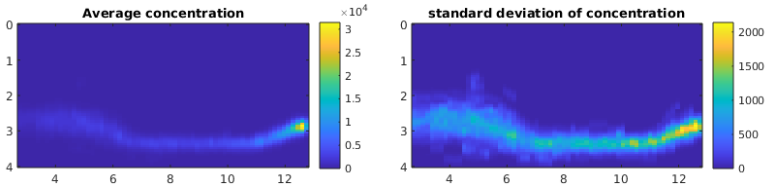
The wind maps are then loaded in Webots to simulate the gas plume generation. For each wind map, 100 simulations were conducted, with the gas source placed randomly, outside any obstacle. In each simulation, a gas concentration map was generated by recording the concentration data using simulated static sensor network placed on a 64×64 grid. Each sensor gathers 5 s of gas concentration data, then both the average and the standard deviation are logged along with the position of the sensor. In total, 18000 gas concentration maps, each of them characterized by 4096 sample points, were gathered to train the plume model.

The plume model is required to take, as input, the source position as well as the environment map and generate, as output, two maps corresponding to the average and the standard deviation of gas concentration. However, in order to make the learning process more efficient, we need to generate feature maps for the input. Inspired by [82] and [83], for each environment map, we create three feature maps.

The geometry of the environment is the most important factor in shaping the the steady-state flow which, in turn, shapes the plume. A steady-state flow is mostly defined by the boundary conditions which includes the properties of each region in the map, for instance, the air flow input, the output, the walls, etc. Therefore, the first feature map used for learning the plume model is dedicated to label each region of the map according to its flow properties. For instance, the free space is labeled 1, the walls 2, the inlet 3, the outlet 4, and the cells inside obstacles 0. On top of flow properties definitions, another feature map should reflect the geometry of the map, which means the position of obstacles. A binary map indicating the presence of obstacles in every cell would not be informative enough as it does not differentiate between cells that are very close to obstacles or the ones in the free space. A Signed Distance



(a) The feature maps



(b) The concentration maps

Figure 14.2 – An example of features maps (a) and concentration maps (b) of the same simulated environment.

Function (SDF) [83], however, provides for every cell in the map, the distance to a surface. Moreover, points that are inside an obstacle will be given negative values to make full distinction between different cell placements. Finally, the model needs the position of the source to shape the plume. Therefore, the last feature map shows, for every cell, the Euclidean distance from the source position. An example of the three feature maps are shown in Figure 14.2a.

Test dataset

The model is expected to perform well in realistic indoor environments. Therefore, in order to evaluate its performance, we designed nine test maps, illustrated in Figure 14.3, with different complexity levels on which we test the performance of the model. The test maps are designed in such a way that, depending on the source position, different plume shapes can be generated. If the obstacles are not large enough, they barely interfere with the air flow and therefore the plume shape will be similar in different settings. On the other hand, if the obstacles are too large, the air flow leads the plume in a narrow passage no matter where the source is placed in the environment. Both of those scenarios produce non-challenging setups for gas source localization. Therefore, the obstacle sizes and positions are chosen proportionally to the problem to be challenging enough for our task.

For each test map, 20 ground truth concentration maps are generated with random source positions. The performance evaluation of the data-driven plume model will be done by calculating the RMSE between the ground truth of the test maps and the output of the trained model for a given source position. The results are presented in Section 14.2.3.

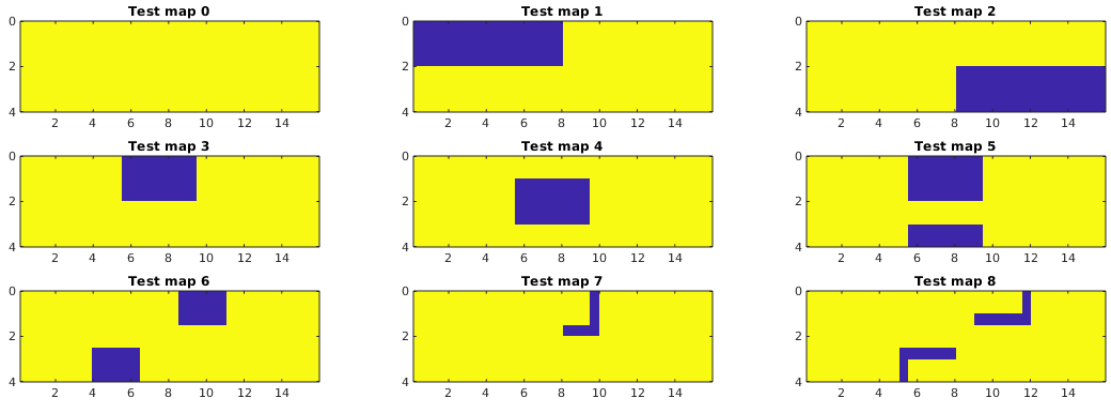


Figure 14.3 – The schematics of the nine test maps.

14.2.2 Convolutional Neural Network architecture

Since the nature of the phenomenon that we seek to learn depends on the geometry of maps, a Convolutional Neural Network (CNN) is leveraged. CNNs are widely used in computer vision and other fields where a strong spatial dependency exists in the data. In recent works on fluid-dynamics surrogate models, such as [83] [84] [85] [86], CNNs showed to drastically improve computational time compared to traditional CFD while resulting in high accuracy.

Since the desired outputs of the network, as defined in Section 14.2.1, are also maps, the architecture of this CNN needs be of type encoder-decoder. In related works, where a CNN is designed to predict many fluid mechanics properties with the same network, a shared-encoding and separated-decoding architecture is leveraged [83] [87] [82].

In [82], in particular, a U-Net architecture is designed to predict wind velocity and pressure, based on the geometry of an environment map. Moreover, all the hyperparameters of the network were optimized through a search between 108 combinations. The resulting architecture is composed of a four-block encoder-decoder, where each block contains three convolution layers followed by a max pooling and a ReLU layer. The kernel size of five was chosen and the number of filters for the four blocks are respectively 8, 16, 32, 32 for the encoder block, and the same, but in opposite order for the decoder blocks. The optimizer is chosen to be AdamW with a learning rate of 0.001, the weight decay of 0.005, and with batch and weight normalizations to be off.

Since the architecture used in [82] was well investigated and the application domain is close to our case, we have conducted the learning process with the same network and parameters along with the provided open-source code.

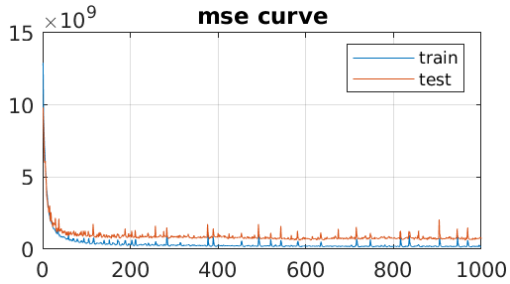


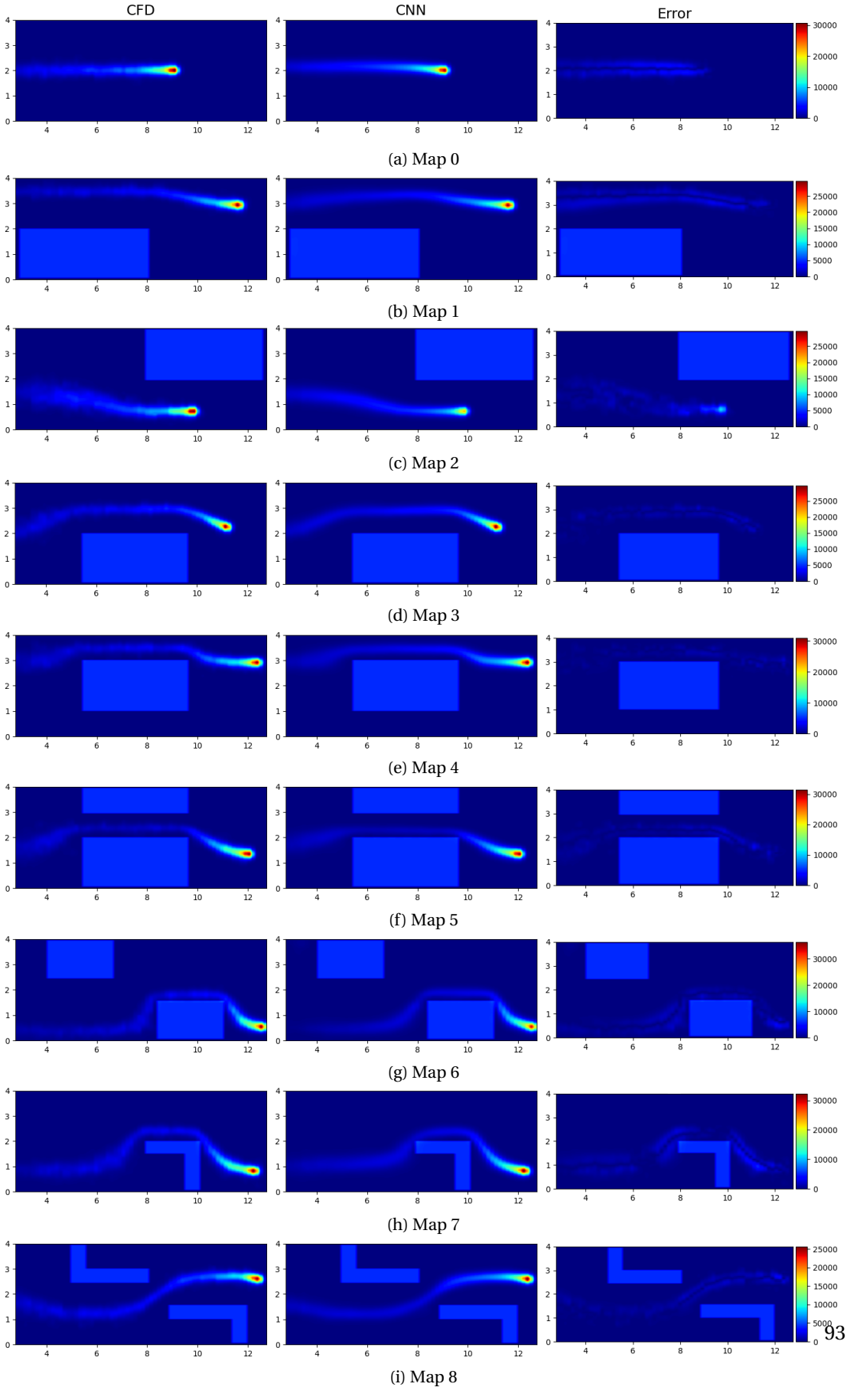
Figure 14.4 – The training loss function over 1000 epochs.

14.2.3 Model Evaluation

Once the dataset and the network architecture are prepared, we have trained the model in 1000 epochs with the 18000 samples, divided in 90% as training set and 10% as test set. The loss function of the network, i.e. the MSE between the ground truth and the output map over all epochs is illustrated in Figure 14.4 with the blue curve for the training set and the orange one for the test set. The value of the loss function decreases rapidly on the first 100 epochs, and continues to slightly decrease until 1000 epochs. The high value of the MSE, i.e. around 0.17×10^9 for training and 0.71×10^9 for test sets, even at the end of the training, is due to the large scale of data in the output. Indeed, the concentration values can go from 0 to more than 3000 for each pixel in the map. A normalization by the maximum value would be possible, but we preferred to not limit the range of obtained concentration, as it depends on many external factors, such as the source release rate and the wind speed, on which the network does not have any prior knowledge.

Once the training is complete, the model is extracted and applied to the nine test maps, shown in Figure 14.3, specifically designed to evaluate the performance of the model, with different source positions. The generated concentration map is then compared to the ground truth. For each test map, an example of plume generated by the model is shown in Figure 14.5. The picture on the right-hand side is the ground truth, the middle one is the concentration map generated by our surrogate model, and the left-hand one is the prediction error. As shown in the maps, the shape of the plume is predicted correctly in every map and the error is negligible.

It is worth noting that the deduced plume is driven only from spatial information and the source location. Including a wind map, instead of the spatial data, would extremely facilitate the prediction, as a plume is physically shaped by the location of the source and the surrounding wind velocity. Therefore, with both these information, it would be trivial to obtain the concentration map. In this case, however, we do not include any wind-related information, apart from the inlet and outlet position. The plume shape is deduced only from the geometry of the environment.



(a) Map 0

(b) Map 1

(c) Map 2

(d) Map 3

(e) Map 4

(f) Map 5

(g) Map 6

(h) Map 7

(i) Map 8

Figure 14.5 – Plume examples generated with our surrogate model.

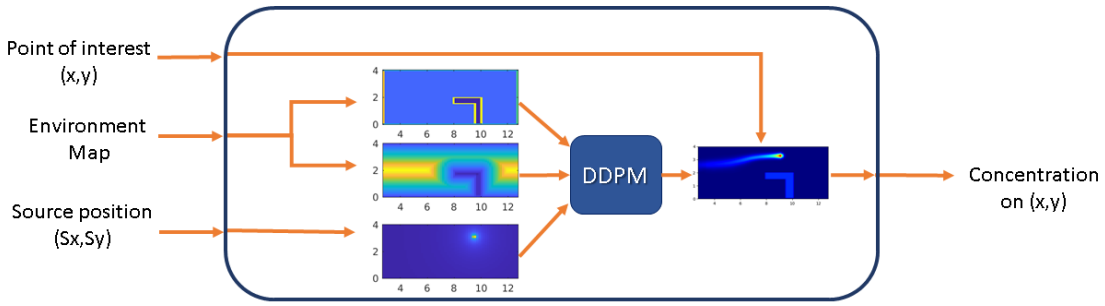


Figure 14.6 – A schematics of DDPM with its input and output.

14.3 Performance Evaluation in the STE Framework

Once the model shows to be successful in predicting the plume shape in environments with obstacles, we integrated it in the STE framework to be evaluated in a GSL application.

14.3.1 Adaptations to cluttered environment

The surrogate model replaces the pseudo-Gaussian analytical equation used in uncluttered areas, presented in Section 7.1. Unlike the previous model, the DDPM has only two parameters which are the coordinates (x, y) of the source position.

To predict the concentration of a location in the map, given a source position, first, our program needs to create the input feature maps. The two spatial feature maps, i.e. SDF and region labels, are only dependent on the map of the environment, which is supposed to be known by the algorithm. Therefore, they are defined and kept fix throughout the experiment. The last feature map, however, indicates the source position, and needs to be created for each source term evaluation. It is done by affecting every cell of the map, the distance of the cell center to the given source position.

Once the feature maps are available, they are fed into the extracted model to generate the concentration map. The concentration values of a set of specific locations are then extracted from the map and given back as output. A schematics of this integration is illustrated in Figure 14.6.

The model generates a concentration model as output, which is efficient for reading the predicted concentration values of many locations at once. Since the computational cost of this model is orders of magnitude higher than the Gaussian plume model, such characteristics help reducing the necessary time to perform prediction calculations.

On top of the model, other adaptations are necessary in the behavior of the robot in cluttered environments. For instance, the robot needs to plan its trajectory from one point to another to not collide with obstacles. Since the map of the arena is known, one can leverage a state-of-

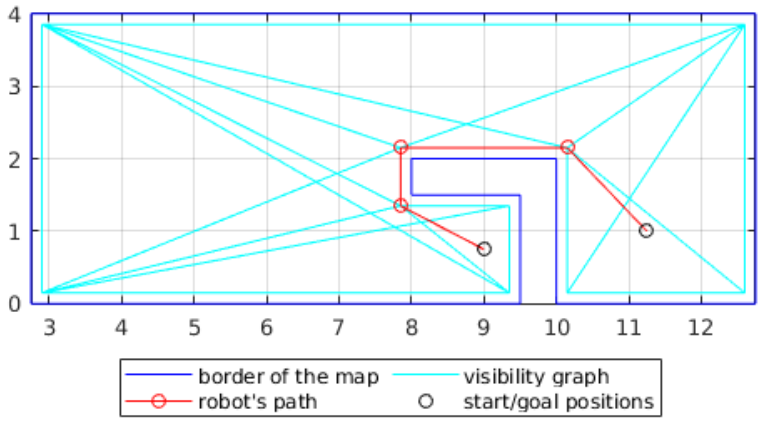


Figure 14.7 – A trajectory example using the VG algorithm.

the-art path planning algorithm. In this thesis, we chose the Visibility Graph (VG) algorithm that allows the robot to travel around the edges of the obstacles to reach its goal position. The algorithm consists on creating a graph where nodes are the start point, the goal point, and the vertices of the obstacles. Then nodes are connected together if a straight line can be drawn from one node to another without being obstructed by an obstacle. This creates a graph where each edge connects two nodes that are visible to each other; hence the name Visibility Graph. Therefore, the robot can travel on this graph using an A* search to find the shortest path.

The implementation of this algorithm was taken from [88] and adapted to the STE framework. However, to avoid the robot hitting the obstacle while finding its path, we have defined the nodes with a 15 cm margin from the walls of the arena and the obstacles. A trajectory example is depicted in Figure 14.7, where a robot, instead of traveling on a straight line if no obstacles were present, goes around the concave obstacle to reach its goal position. The trajectory, while not being the smoothest path possible, is the shortest one and guarantees a safe route to the destination.

14.3.2 Simulation experiments

We evaluated the integration of the DDPM in our STE framework in simulation experiments, with all nine test maps. The statistical results on the estimation error are depicted in Figure 14.8. Overall, the method shows to be successful in overcoming cluttered environments and localize the source using the surrogate model. More outliers can be seen in the results compared to the uncluttered environment, which is due to the higher complexity of the task.

Comparing the performance of the surrogate model and the Gaussian plume model, i.e. map 0 and the results presented in Figure 13.3, we observe a lower number of iterations for the DDPM. However, this is due to the fact that the surrogate model was trained in the same simulation setup as the evaluation. In other words, the DDPM is based on the filament-based plume model. Therefore, comparing its performance with another plume model, which is not

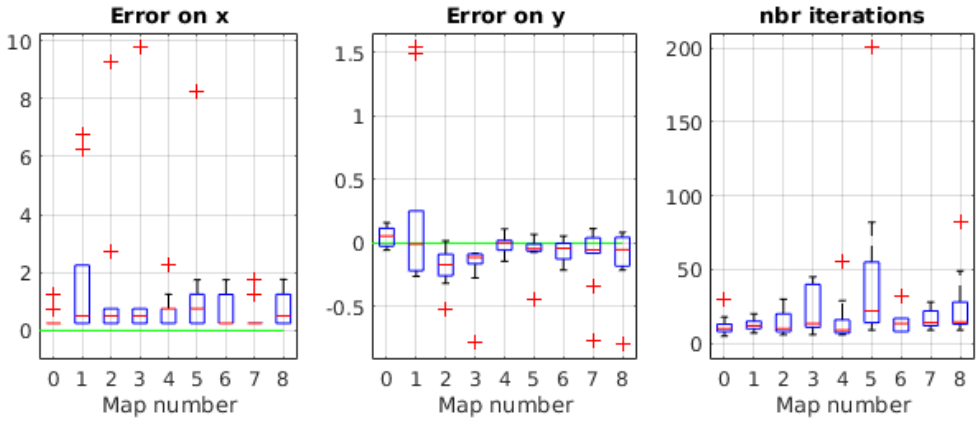


Figure 14.8 – Statistics of simulation results.

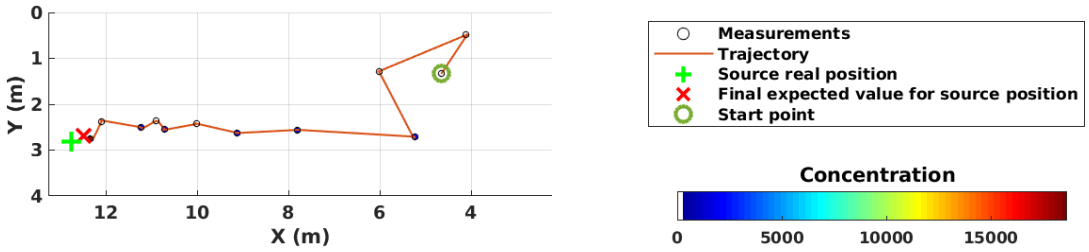
based on the same plume model, appears to be unfair.

Examples of trajectories are presented in Figure 14.9 for each test map. The robot manages to find its way toward the source while avoiding obstacles. Compared to previous evaluations with the Gaussian plume model shown in Figure 13.2, the trajectories seems to be smoother with less measurements and longer steps.

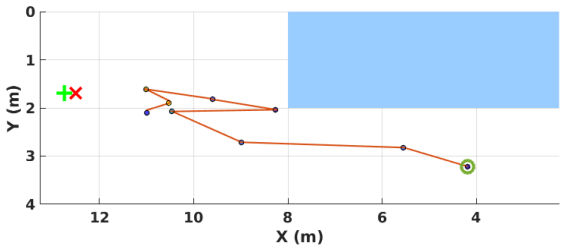
14.4 Conclusion

We successfully created a surrogate plume model based on simulated concentration data with random cluttered environments. The model was trained using a CNN and was evaluated with nine test maps of various complexity. The model was then integrated in our STE framework, where it was evaluated in the test maps with a mobile robot. The results showed smoother trajectories with fewer iterations compared to our core STE algorithm presented in Chapter 7.

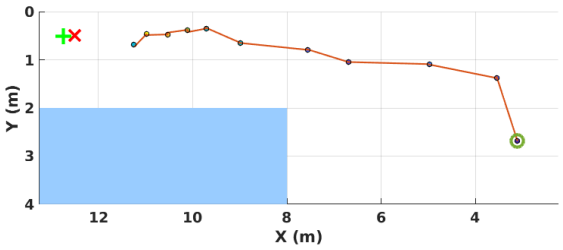
Experiments in physical reality are necessary to assess the performance of the model and its integration in the STE framework. Moreover, the model can be extended to take more parameters as input, such as wind velocity and source release rate.



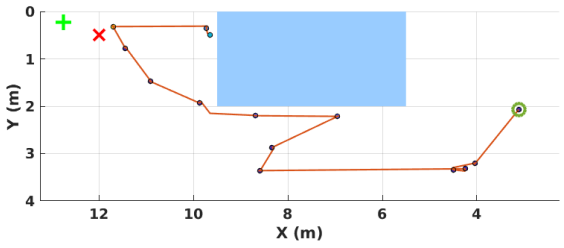
(a) Map 0



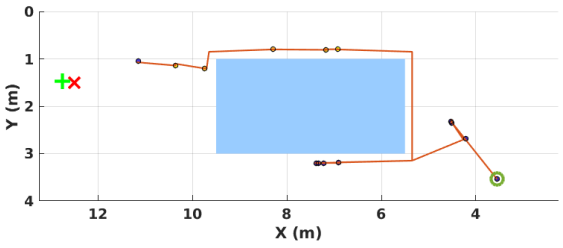
(b) Map 1



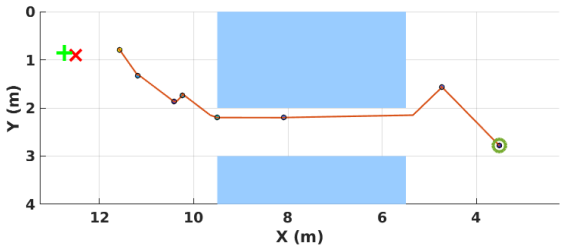
(c) Map 2



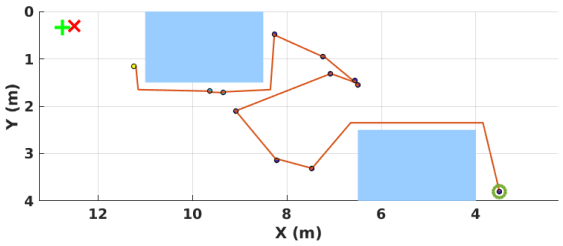
(d) Map 3



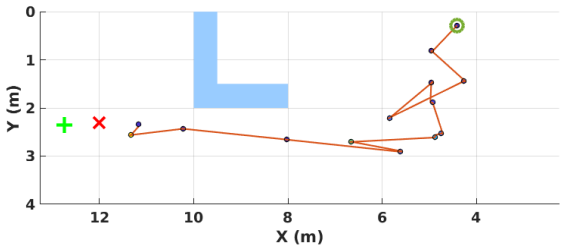
(e) Map 4



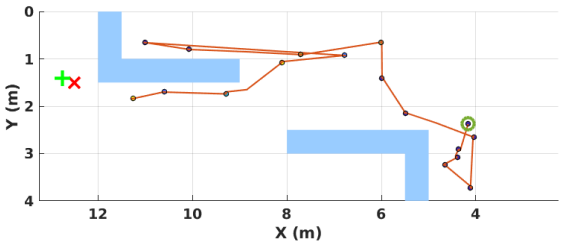
(f) Map 5



(g) Map 6



(h) Map 7



(i) Map 8

Figure 14.9 – Trajectory examples for each test map.

Benchmarking the STE Algorithm Part V

15 Introduction

After validating an algorithm, it is important to compare its performance with other methods from state of the art. In the literature, often new algorithm are compared to bio-inspired methods (e.g., [89] [29] [90]), since they are the most studied and well-established class of algorithms for GSL problems.

In this part of the thesis, we compare the core STE method presented in Chapter 7 and its 3D variation presented in Chapter 11 with two bio-inspired algorithms. The bio-inspired method are themselves benchmarked with the most studied and well-known methods in the literature, namely the moth-inspired algorithms.

Since the STE algorithm presented in this thesis is a probabilistic method and has major differences with bio-inspired algorithms, the comparison is non-trivial. Therefore, before entering into the details of performance comparison, we briefly present the qualitative differences of the two types of methods and we explain the conditions and metrics we use to make a fair quantitative comparison.

15.1 Qualitative Comparison

Before quantitatively measuring the performance of our proposed method compared to the bio-inspired algorithm, we need to mention the major technical and theoretical differences in the two types of algorithms, bio-inspired and STE.

- *Scope and starting position* - The most fundamental difference is that the bio-inspired algorithms often tackle only one sub-task of the GSL problem. For instance, moth-inspired algorithms address only plume tracking, which means the robot always starts inside the plume and tracks the samples to the source. Whereas in the STE, all three subtasks are addressed at the same time with the same method, and the robot can start anywhere in the environment.

- *Source declaration* - For bio-inspired algorithms, the source is declared as found, as soon as the robot enters a certain area around the source, which could be intentional or unintentional. While with STE, no matter how close the robot is to the source, it only declares the source position when it has enough information to do so. It could also declare the source far from it, based on the information gathered in the field. This is usually not observed in our method, since we specifically guide the robot towards the source, but it is possible that the final sample is taken far from the source.

- *A priori information* - One major advantage of bio-inspired algorithms is that they need no a priori information on the environment. Since they only follow local wind and gas measurements toward the source, they can be deployable in unknown environments. Whereas the general form of our STE algorithm depends on the map of the environment. Although we presented a variation of STE that does not rely on environment map in Chapter 12, here, we make the comparison with the core method of this thesis, presented in Chapter 7.

- *Adaptation to environmental conditions* - For bio-inspired algorithms the gas plume is a binary field, i.e. the robot is either inside or outside the plume. This binarization is made using a threshold value. If the robot senses a concentration value higher than the threshold, it supposes to be inside the plume and outside otherwise. Therefore, the value of the threshold has a major impact on the shape of the binary plume, which in turn, has a significant effect on the robot's behavior and hence on the performance of the method. Since the ideal value depends on the plume characteristics, this value is usually set arbitrarily, which prevents the algorithm to be successful in different environmental conditions with the same performance. STE, however, relies on a plume model and thus, as long as the plume model is valid in an environmental condition, so is the algorithm.

- *Strategy* - Moreover, the strategy of the two methods are fundamentally different, since in bio-inspired algorithms, the robot follows the wind and gas concentration cues, while in STE the robot looks for more informative points, which could be inside or outside the plume. For instance, even sampling a zero concentration value can exclude some positions in the environment from containing the source. However, high concentration points are relatively more informative, since they give a direct information on the source location and thus exclude many other points that do not contain the source.

- *Success measurement* - Another major difference between the two methods is related to performance evaluation. For bio-inspired algorithms, two well-known metrics are usually used: success rate and distance overhead. The success rate is a percentage of successful runs. When the robot reaches the source, it ends the experiment in a success. A run is considered unsuccessful if the robot hits the wall of the arena or runs out of a given time (or iteration number) before reaching the source. The distance overhead is the ratio between the traveled distance over the shortest distance from the starting position of the robot to the source. For STE, however, the success is measured by comparing the expected value of the belief on the source position and the ground

truth. Therefore, for STE the success is measured in meter. We often refer to the traveled distance or number of iterations to show the difference in computation or energy consumption between different setups. Moreover, in STE, there is no criterion for calling a run unsuccessful. There are expectational cases where the error is very high, but in the majority of runs, it is not the case.

15.2 Quantitative Comparison

Given all the difference mentioned above, to make a fair comparison between STE and a bio-inspired algorithm, we need to establish suitable conditions for both methods. The following conditions were applied for this comparison:

- In case where the bio-inspired algorithm is only a plume tracking method, the robot should be inside the plume at the beginning of the experiment.
- Hitting the walls of the arena does not end the experiment. This is for the sake of fairness, since the STE algorithm uses the map of the environment as a priori information, which is not the case for bio-inspired methods.
- Entering a circular area of a fixed radius around the source ends the experiment in success. In STE, it is possible that the robot declares the source before entering this radius, in this case, it is also considered as success.
- Exceeding the maximum number of iterations, the maximum amount of time, or the maximum traveled distance result in a failed run.

Additionally, the right metrics should be used to highlight the important characteristics of the two methods. The success rate could be used in our comparison, since we established rules to declare a run successful or failed. However, we are making the comparison with algorithms that showed to be successful almost in 100% of runs in simulation. Therefore this metric will be irrelevant to show. Moreover, the distance overhead does not seem a suitable metric in this comparison. Indeed, distance overhead is designed for methods that lead the robot straight to the source, whereas the STE algorithm is not designed in this way. It needs to gather information, and this means that it needs to sample different locations to gather enough information on the source position. On the other hand, traveled distance is also a measure of spent time in the experiment, which makes sense for both STE and bio-inspired algorithms, since the objective is always to localize the source in the shortest time possible.

Therefore the metric used in this comparison will be the traveled distance, which is the distance that the robot travels from starting point to reach the source.

In the following chapters, the STE algorithm will be compared in terms of performances with different methods in various scenarios.

16 Benchmarking in a 2D Search

16.1 Introduction

The first scenario in which the comparison is taking place is the simplest one. We consider a 2D uncluttered space with a quasi-laminar and constant airflow. The STE method and the benchmark algorithm will be evaluated with different wind speed and source release rate in simulation.

The chosen benchmark method is the novel bio-inspired algorithm we published in [91] where the algorithm was evaluated in different environmental conditions in the wind tunnel and was compared to three well-established moth-inspired algorithms. The method was shown to be more robust in the tested environmental conditions compared to the moth-inspired ones. This is why we chose this method as benchmark instead of the well-known moth-inspired algorithms, such as casting, surge-cast, and surge-spiral.

16.2 Benchmark Algorithm: Adaptive Lévy Taxis

Unlike moth-inspired algorithms which are directly inspired from one particular species, Lévy Taxis tries to reproduce what can be observed at many scales among a large variety of living beings [92]. Originally designed to address the plume finding phase, it is particularly interesting for its relatively low complexity and its potential to dynamically adapt to the environment to face discontinuous and peaky plumes, thanks to its intrinsic stochasticity. We based our work on this algorithm to push it further and extend its environmental adaptability by considering gas concentration gradients. We thus present a new algorithm called Adaptive Lévy Taxis (ALT), which is based on the Lévy Taxis (LT) algorithm presented in [9]. ALT differs from LT as the movement decision at each step is based on the measured gradient of the plume. This allows the agent to continuously adapt the walk to the environment and thus better address the plume tracking phase. The advantages of such an algorithm are: its independence from absolute concentrations (which are usually affected by sensor's drift in time), its computational simplicity, and its suitability to address the plume tracking problem.

16.2.1 Algorithm

In this section, we will first briefly introduce the LT algorithm, before describing ALT.

Lévy Taxis

In 2009, Pasternak et al. presented LT [9] as a novel algorithm for finding a chemical plume (i.e. *plume acquisition* phase). It is a search strategy of the same category as Correlated Random Walk (CRW) [93] and Lévy Walk (LW) [92]. These are randomized search algorithms that share the same global process: the agent starts from a random point and has to move in the space, according to a governing law, in order to find the chemical plume. At each step, the robot should walk to a point determined by a move length (M_l) (AKA step length) and a turning angle (T_a), which are both calculated based on a fixed probability distribution. The main differences between these search strategies are the choice of the probability distributions determined by the value of their key parameters.

LT combines the M_l probability distribution of LW and the T_a probability distribution of CRW. Hence M_l and T_a of each step are calculated using Eq. (16.1) and Eq. (16.2),

$$M_l = L_{min}.r^{\frac{1}{1-\mu}} \tag{16.1}$$

with r a random variable uniformly distributed $r \in [0,1]$, L_{min} the minimum move length, and μ (Lévy-index) the move length's key parameter ($1 < \mu \leq 3$).

$$T_a = \left[2 \cdot arctan\left(\frac{1-\gamma}{1+\gamma} \cdot tan(\pi \cdot (r-0.5))\right)\right] + bias \tag{16.2}$$

Where r is a random variable uniformly distributed $r \in [0,1]$ and γ the turning angle's key parameter that shapes the distribution ($0 \leq \gamma \leq 1$). The $bias$ is the upwind angle: this brings the center of the distribution at the upwind direction, to make the latter the most probably chosen value. Table 16.1 presents the governing laws of the algorithms presented in [9] as well as the values of their key parameters.

The value of the key parameters μ and γ are chosen before the start of the algorithm and they are kept constant during a run. However, the values of M_l and T_a are calculated at each step after their respective random parameters r are picked. Once a new point in the space has been chosen to be sampled, the agent walks towards it with a constant speed.

All these search algorithms have been evaluated in [9] using a chemical plume data set for different pairs of the key parameters. The performance of each setup has been assessed by measuring the detection success rate (i.e. percentage of successful runs over a total of 50 runs) as well as the path directness (i.e. the percentage ratio of the shortest distance between the start point and the plume detection point over the total traveled distance).

Table 16.1 – Governing laws and key parameters of random walks (adapted from [9]).

	Move Length (M_l)	μ	**Turning Angle** (T_a)	γ
Brownian Walk	Asymptotically Gaussian-like	3	Uniform	0
Lévy Walk	Power Law	$1<\mu\leq 3$	Uniform	0
Correlated Random Walk	Asymptotically Gaussian-like	3	Wrapped Cauchy	$0\leq\gamma\leq 1$
Lévy Taxis	Power Law	$1<\mu\leq 3$	Wrapped Cauchy	$0\leq\gamma\leq 1$

Comparing the performance of all these algorithms together shows that both LW and LT yield the best detection success rate ($\approx 85\%$) with $\mu = 2.8$ and both CRW and LT give the best path directness ($\approx 10\%$) using $\gamma = 0.05$ [9]. This implies that LW have the best success rate but tends to travel more to find the plume, compared to CRW. On the other hand, CRW finds a more direct path to the plume, but has a lower detection success rate. Yet, as LT cleverly combines their governing laws, it brings together the best success rate obtained from LW and the shortest traveled distance from CRW.

In addition to its remarkable performance, LT has the advantage of being environmentally adaptive: since it uses the upwind angle as a bias for the navigation, the strategy is continuously adapted to the environment of the experiment. These two significant assets of the LT algorithm have been the main inspirations for our previous contribution [91]: using LT to address the plume tracking phase by introducing chemo-taxis and thus designing the new ALT algorithm.

Adaptive Lévy Taxis

As mentioned earlier, LT has the advantage to adapt the search operation to the environment by orienting the agent to the upwind direction. However, the stochastic nature of this algorithm prevents it from being successful in the plume tracking phase, using only the wind direction as bias. An agent performing LT in this phase would constantly go out of the plume just after finding it and thus, would have to restart the search for the plume all over again. Indeed, LT does not perform chemo-taxis as its movements are not influenced by chemical cues. In order for the agent to stay in the plume once it is found, this type of algorithms needs to adapt the walk to the concentration as well, so it can find its path to the source. Thus, we propose to dynamically change the key parameters (i.e. μ and γ) of the ALT algorithm, according to the gas gradient measured. This modification enables ALT to address the chemical plume tracking problem.

At each step, the gas concentration is measured and compared to the one of the previous step to calculate the local gradient ∇C between the current sampling position and the previous one. We then use the value of the local gradient to tune the key parameters of the governing

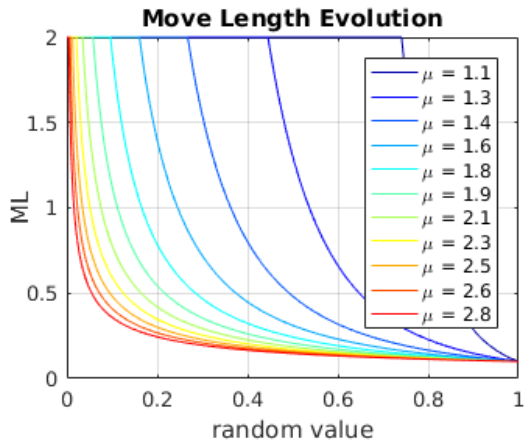


Figure 16.1 – M_l distribution according to its governing parameter $\mu \in [1.1; 2.9]$. M_l itself varies between 0.1 and 2.0.

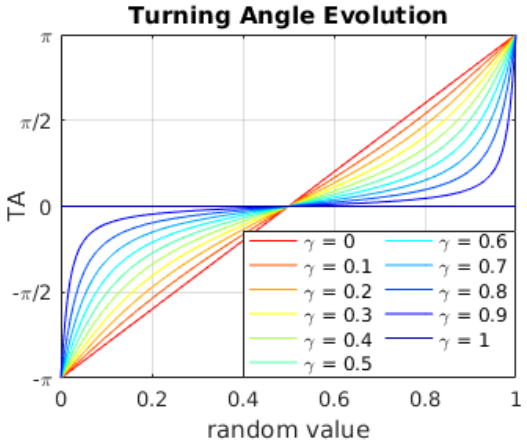


Figure 16.2 – T_a distribution in function of $\gamma \in [0; 1]$. $\gamma = 0$ yields a uniform distribution while $\gamma = 1$ yields $T_a = 0$ anyway.

laws before taking a new step – instead of keeping their value constant over the run which is the case in the original Lévy Taxis algorithm developed in [9].

In long-term exposures (i.e. in the case where the robot moves slowly and each sample is the average of many concentration measurements), when the average local gradient is strongly positive, it theoretically means that the robot is closer to the source compared to the previous step, so the next step can be a long move straight towards the upwind direction, i.e. exploitation. On the other hand, in the case where the gradient is around zero, as the agent does not get any strong clue about the variation of the plume, it takes a short move in a random direction in order to promote exploration. The former case corresponds to key parameters μ and γ tending to 1, while in the latter μ tends to its maximal value 3 and γ to its minimal value 0.

Equation (16.3a) and Equation (16.3b) show the relationship between the gradient of gas concentration ∇C and the key parameters μ and γ, while Figure 16.1 and Figure 16.2 present how the latter influences the probability distribution of M_l and T_a.

$$\mu = \begin{cases} \mu_{min}, & \text{if } |\nabla C| > \theta \\ \mu_{max} - \frac{|\nabla C|}{\theta}(\mu_{max} - \mu_{min}), & \text{otherwise} \end{cases} \tag{16.3a}$$

$$\gamma = \begin{cases} \gamma_{max}, & \text{if } |\nabla C| > \theta \\ \gamma_{min} + \frac{|\nabla C|}{\theta}(\gamma_{max} - \gamma_{min}), & \text{otherwise} \end{cases} \tag{16.3b}$$

Where ∇C is the gradient of gas concentration and θ the threshold beyond which the values of μ and γ remain at μ_{min} and γ_{max} respectively. These boundaries were set based on Pasternak's work [9].

Besides adapting key parameters during the run, the gradient serves another purpose: as it is

calculated after taking one step, it brings information about the rightness of the last decision on the turning angle T_a. Therefore, the heading of the robot can be used as a complementary information to the wind direction for the bias used in the calculation of T_a. For instance, if after taking one step the gradient is positive, which means that the last direction was the right decision, the angle for the next step should be biased not only by the local wind direction but also by the forward direction. Thus, in ALT, the bias for T_a is taken as an average of the heading of the robot and the upwind angle, unlike LT which only takes into account the local wind direction. This acts as a low-pass filter that prevents the heading from varying too much in one step.

However, in case the gradient is negative, which means the last chosen direction was not a good one (the agent is probably not heading towards the source), only the upwind angle is used as bias. Therefore the agent tends to move upwind to encounter a higher gas concentration.

Algorithm 2: Pseudo code of the Adaptive Lévy Taxis algorithm.

```
while (step < stepLimit) do
    Sample gas concentration (c_c) and read current position (p_c);
    Calculate gas concentration gradient ∇C = c_c − c_p;
    Adapt parameters μ and γ using Eq. (16.3a) and Eq. (16.3b);
    Calculate M_l and T_a using Eq. (17.1) and Eq. (16.2);
    Apply M_l and T_a to p_c to get next goal (G_x, G_y);
    previous gas concentration (c_p) ← current gas concentration (c_c);
    previous position (p_p) ← current position (p_c);
    while ((G_x, G_y) is not reached) do
        Move towards (G_x, G_y);
        if (Source is found) then
            Declare success and Terminate algorithm;
        end
        if (Wall is encountered) then
            Declare failure and Terminate algorithm;
        end
    end
    step ← step + 1;
end
Declare failure;
```

The pseudo-code above provides an overview of the ALT algorithm. In this algorithm, at the beginning of each step, the position of the agent as well as the local wind direction and the gas concentration are logged. Then the concentration gradient is calculated using the last step's log. Using this information, M_l and T_a are calculated to determine the goal towards which the agent has to walk. Once the source is found, which is determined thanks to an external input, the agent declares success and terminates the algorithm. In case the agent reaches the limits of the experimentation area, it declares failure and terminates. On top of that, we used a

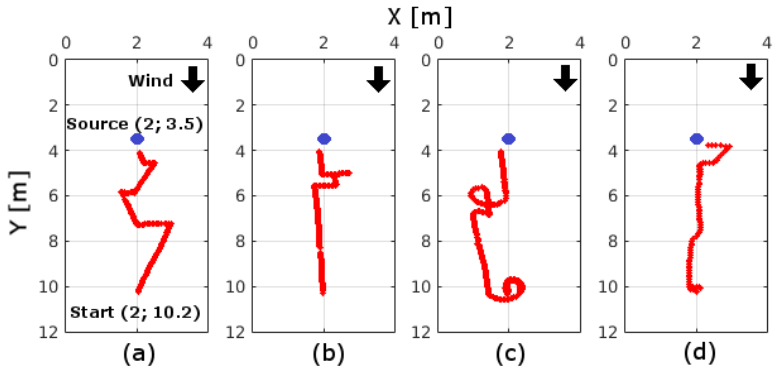


Figure 16.3 – Examples of trajectories obtained in the wind tunnel with a wind speed of 1.0 m/s and a release rate of 10%: (a) Casting, (b) Surge-Cast, (c) Surge-Spiral and (d) Adaptive Lévy Taxis algorithm.

limit for the number of steps, for the agent to avoid spending too much experimentation time wandering far from the source. If this steps limit is reached, the algorithm also declares failure.

16.2.2 Performance evaluation

In the original paper, the ALT method, as well as the three moth-inspired algorithms, were evaluated in our wind tunnel in wind speeds ranging from 0.1 to 1.0 m/s and with source release rates from 10 to 50% of the pump capacity. Example of successful trajectories for all four algorithms are shown in Figure 16.3. The statistical results are illustrated in Figure 16.4.

The average distance overhead in all setups was shown to be lower than 1.2 (i.e. close to 1.0 that is the best possible value). In addition, ALT showed satisfactory results ($\geq$ 5/10) in terms of success rate in all configurations. The consistency of these good results in all the setups is a significant performance, not achieved by moth-inspired algorithms.

All three moth-inspired algorithms have remarkable performances in setups with high release rate (50%) and high wind speed ($\geq$ 0.5 m/s). In such conditions, where the source emits a fair amount of gas filaments and the wind folds the plume in the center, it is easy to follow the plume. However, Casting and Surge-Cast showed to be very sensitive to extreme environmental conditions, for instance, when the wind speed was low (0.1 m/s) or when the release rate of the source was set as low as 10%. The Surge-Spiral algorithm, unlike the two others, yields outstanding success rates in all the setups. In case of difficult setups (e.g., wind speed = 0.1 m/s and release rate = 10%) spiraling has always a satisfactory success rate ($\geq$ 7/10), but a relatively high distance overhead. In fact, the spiraling strategy helps the robot to eventually re-acquire the plume after losing it. However, it also implies taking more steps, compared to the two previous algorithms, which leads to a higher distance overhead.

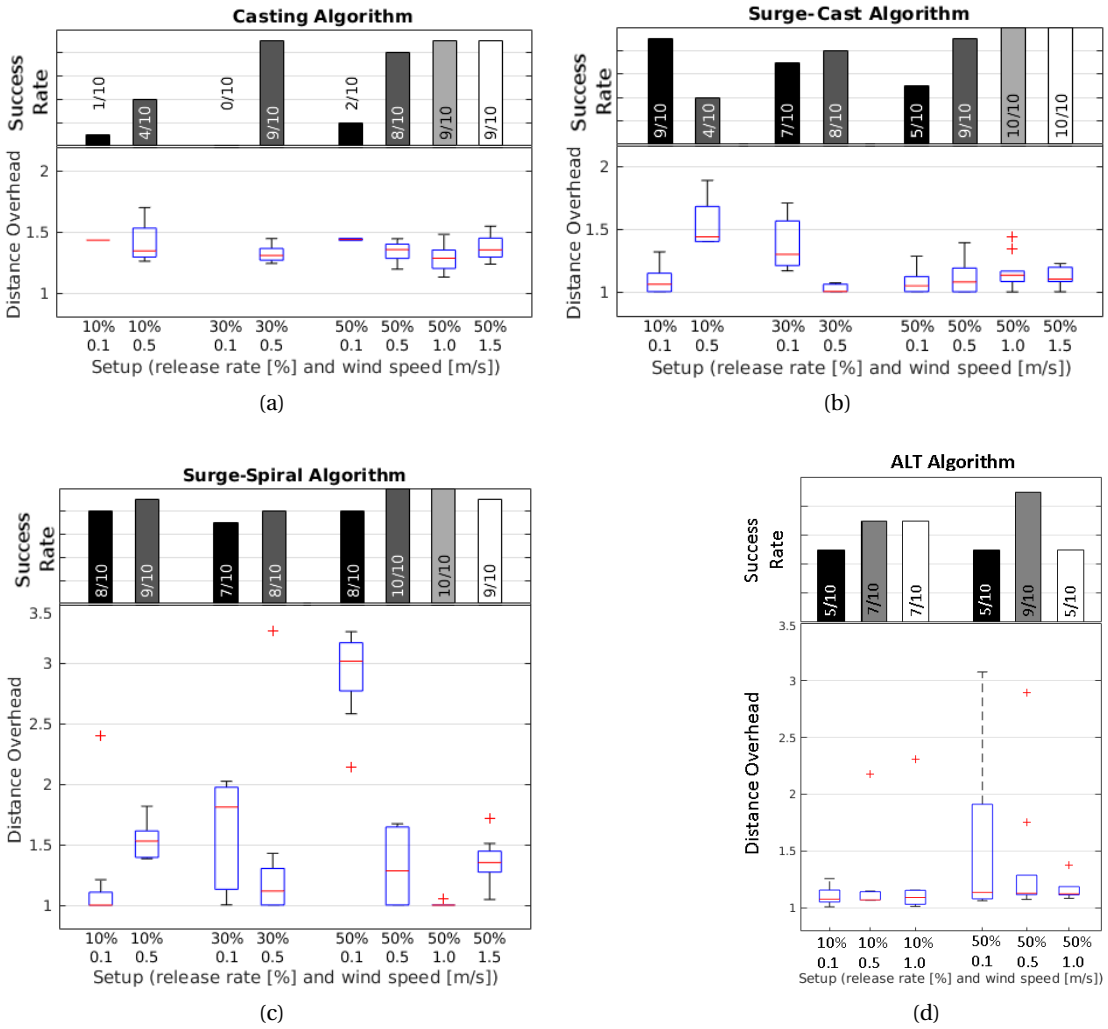


Figure 16.4 – Performances of Casting (a), Surge-Cast (b), Surge-Spiral (c), and ALT (d) algorithms in the wind tunnel.

16.2.3 Conclusion

In case of a wall free arena, a robot performing ALT, could eventually be able to reacquire the plume after losing it (yielding to 100% success rate), thanks to the stochastic search properties of this algorithm. Indeed, convergence in 3D was demonstrated in [94] for a Lévy Flight. In contrast, moth-inspired algorithms, such as Casting or Surge-Cast, once they lose the plume, have severe difficulties to re-acquire it. The only moth-inspired competitive algorithm from this perspective is the Surge-Spiral, an algorithm which would be also 100% successful in case of a wall-free arena at the price of a very high distance overhead. Probably, only a smart combination of casting, surging, and spiraling, as it actually happens in real moths, would be competitive with our ALT algorithm in all different environmental conditions.

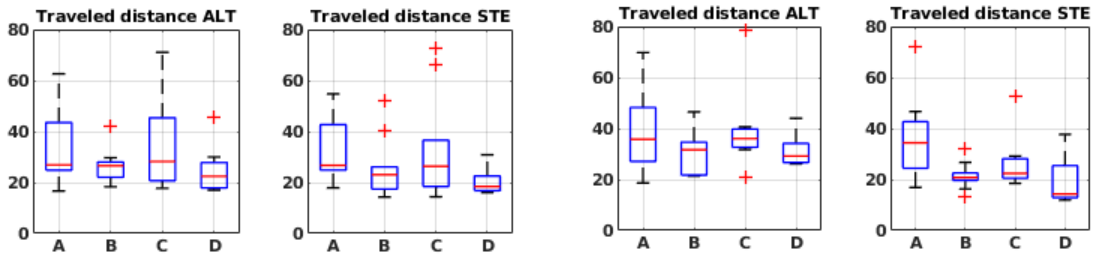
16.3 Performance Comparison with the STE Algorithm

To make the comparison as fair as possible between two methods that are natively different, we introduce slight modifications in start and stop conditions as explained in Chapter 15.

Since ALT is a plume tracking algorithm, we carry out the first set of benchmarking experiments with the starting position of the robot inside the plume. The experiment stops if the robot enters a circular area of 50 cm radius around the source, or, in case of STE, when the source position is declared.

We evaluated both methods in the different environmental conditions reported in Table 8.1. The results illustrated in Figure 16.5a show that the traveled distances with both methods are very similar, with slightly lower medians for STE.

If the robot starts outside of the plume, it does not make a big difference for STE, but it does for ALT. To show this with statistical results, we carried out a second set of experiments, with the same comparison for a setup where the robot is placed randomly on the crosswind axis, therefore potentially inside or outside the plume. The results are presented in Figure 16.5b. The performance of STE indeed shows no significant difference with Figure 16.5a, which is not the case for ALT.



(a) (b)

Figure 16.5 – Performance comparison between ALT and STE when the robot starts (a) inside the plume and (b) on a random position on Y-axis.

17 Benchmarking in a 3D Search Space

17.1 Introduction

As demonstrated in Chapter 11, our STE method was evaluated in a 3D search space in a simulated environment. In order to show its efficiency, we make a comparison between the performances of STE in 3D with the ones of a 3D bio-inspired algorithm. Similarly to the previous chapter, both methods will be evaluated in different environmental conditions in simulation.

The benchmark method, published in [95], is a novel bio-inspired 3D algorithm involving cross-wind Lévy Walk (LW), spiralling and upwind surge. The algorithm has been validated in simulation and evaluated in our wind tunnel, under different conditions in terms of wind speed and source release rates. Studying success rate and execution time, the results showed that the method outperforms its 2D counterpart and it was robust to the various environmental conditions.

17.2 Benchmark Algorithm: 3D Surge-Spiral

Although most of the previous works on GSL are based on wheeled robots in 2D, recently there is some additional attention towards 3D plume tracking. Since bio-inspired algorithms are the most studied class of algorithms in GSL, the adaptation to 3D started with this class as well, with [75] where a robot with a sensor head capable of vertical movements performed an algorithm based on zigzag path of the dung beetles to track the plume in 3D. Moth-inspired algorithms have also been adapted to 3D recently by Gao et al. [76] in simplified simulations, based on traditional 2D techniques. Having multiple sensors on the sides of the robot and measuring the local 3D gradient of gas concentrations, they have added a pitch angle to the heading of the robot. This modification is done only in the phase of upwind surge but not in the phase of plume retrieval. In another work, Edwards et al. [96] proposed a 3D moth-inspired algorithm that uses the plume edge (or the plume center-line in the modified version) to modify the timing of the crosswind movements. This algorithm was tested in a small (about

1 m) realistic environment, using an ion source.

The presented benchmark algorithm addresses both plume acquisition and tracking in a 3D space through a moth-inspired method.

17.2.1 Algorithm

This algorithm covers the two first phases of the GSL problem, i.e. *plume acquisition* and *plume tracking*. In this section, we will present the detail of the algorithm on both sub-tasks.

Lévy Walk in 3D

In the *plume acquisition* phase, the goal being to find the gas plume, the sensor of the robot has not yet detected any gas concentration. Therefore, the robot needs to randomly or systematically scan the environment to find the first indication.

For this phase, we have chosen to adapt LW into 3D, since it has been shown that the probability of returning to the previous position with LW is smaller compared with other random walk mechanisms [97]. LW is performed by many living beings (e.g., Drosophilia [98] and honey bees [99]) for foraging, and has the advantage of allowing the searcher to maximize the number of visited targets, versus the traveled distance [99].

As we are addressing the problem of GSL in 3D, searching the entire environment for the plume seems very time consuming, because the agent does not have any a priori information about the plume. On the other hand, performing a search on a 2D plane perpendicular to the wind direction allows to find the plume in a quicker and more efficient way.

Assuming that the X-axis is along the upwind direction and therefore the crosswind plane would be that defined by the Y-Z-axes, the global process of the first phase of our algorithm is as follows: the agent starts the search from a random position on the crosswind plane, and moves on this plane according to the governing law of LW, in order to encounter the plume. At each step, the robot moves to a point determined by a move length M_l (AKA step length) calculated based on a fixed probability distribution presented in Eq. (17.1), as well as a random turning angle $T_a \in [0, 2\pi]$.

$$M_l = L_{min}.r^{\frac{1}{1-\mu}} \tag{17.1}$$

with r a random variable uniformly distributed $r \in [0,1]$, L_{min} the minimum move length, and μ (Lévy-index) the move length's key parameter ($1 < \mu \leq 3$). In this work, in order to maximize the exploration, we set μ to its maximal value 3 and a relatively long L_{min} of 50 cm.

Thus, if an agent is located at (x_0, y_0, z_0) and has to take a step, once M_l and T_a has been

calculated, the target position to move towards will be as presented in Eq. (17.2).

$$\begin{aligned} x &= x_0 \\ y &= y_0 + M_l \,.\, \sin(T_a) \\ z &= z_0 + M_l \,.\, \cos(T_a) \end{aligned} \tag{17.2}$$

3D plume tracking through upwind surge and spiralling

Once the plume is encountered, the *plume tracking* phase begins. We design two alternating behaviors to address this phase; “upwind surge" and “spiralling on the 3rd dimension". The details of the two behaviors are explained in the following.

In nature, although the behavior of different species (e.g., crab, lobster, moth, salmon) are not exactly the same, they all seem to pursue the upstream path while in contact with the plume [100]. This behavior is used in many bio-inspired robotic GSL works as well (e.g., [76], [75]). Since this strategy seems to be sufficiently efficient, especially in a quasi-laminar flow, which is the case of this work, we have chosen to use it in the second phase of our method. Therefore, when the agent finds itself in the plume, it goes upwind with a constant speed and this movement continues while it considers to still being inside the plume.

However, if the agent loses the plume, either due to the plume’s conic shape or patchiness, or an error in the wind direction estimation, it needs to search the environment in order to reacquire the plume once again. This search should not necessarily be a random walk as in the first phase, since the agent is most probably very close to the plume and thus a local search would be more efficient to find the plume back.

For re-finding the plume, in similar works (e.g., [75, 76]), different bio-inspired behaviors have been developed. One of the behaviors that has always shown (although in 2D) outstanding performance in terms of success rate is the spiralling movement [33]. Therefore, we have decided to take advantage of this particular algorithm and upgrade it to 3D (which is closer to the reality in nature) to address the problem. This behavior is inspired by the spiralling movement of moths while searching for a gas plume [101]. Many studies have implemented a similar behavior but in 2D robotic experiments [33, 102] in highly simplified environmental conditions.

Most moths’ behavior being in 3D, the spiralling movement is performed on a crosswind plane where the chance of plume reacquisition is the highest. So in this work, we have implemented the spiralling movement on the crosswind Y-Z plane. While the spiralling behavior itself is 2D, it helps the agent to find the plume in the right altitude and to continue the upwind surge. Ultimately, all three dimensions are covered by the combination of the upwind surge (along X-axis) and the spiralling (on Y-Z plane). Thus, in this local search, the coordinates of a target

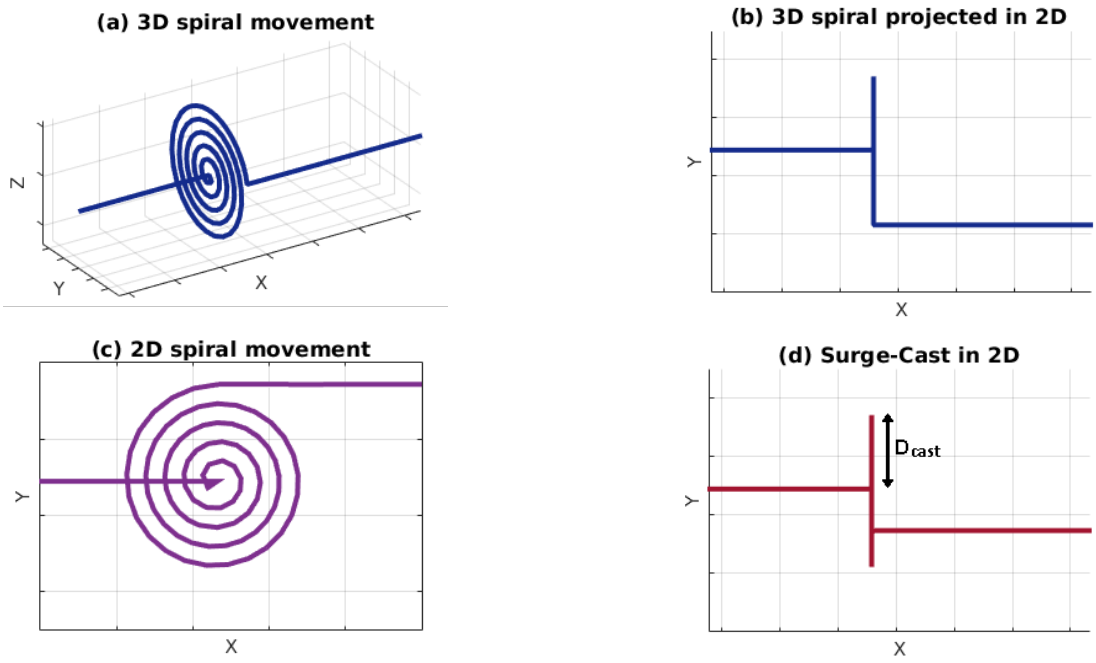


Figure 17.1 – Trajectories of different behaviors: (a) and (b) 3D spiralling, (c) 2D spiralling, (d) 2D Surge-Cast.

position after a step are expressed in Eq.(17.3).

$$\begin{aligned} x &= x_c \\ y &= y_c + s_d \,.\, t \,.\, \cos(2\pi t) \\ z &= z_c + s_d \,.\, t \,.\, \sin(2\pi t) \end{aligned} \tag{17.3}$$

with (x_c, y_c, z_c) the position where the agent loses the plume and thus needs to search the surroundings, s_d the spiral drift which determines the distance between the rounds of the spiral movement, and t the iteration of the local search. As shown by the Eq. (17.3), the movement on the X-axis stops until the plume is retrieved.

Despite the similarity, the projection of this method in 2D would not be the Surge-Spiral, but the Surge-Cast algorithm [33] which performs a local search on the crosswind dimension (see Figure 17.1).

Source identification

In this work, as we focus on the two first phases of the problem, we assume the source is localized if the agent reaches an area of $20 \times 20 \times 20\,\text{cm}^3$ around the source while being still in the plume (see Figure 17.2).

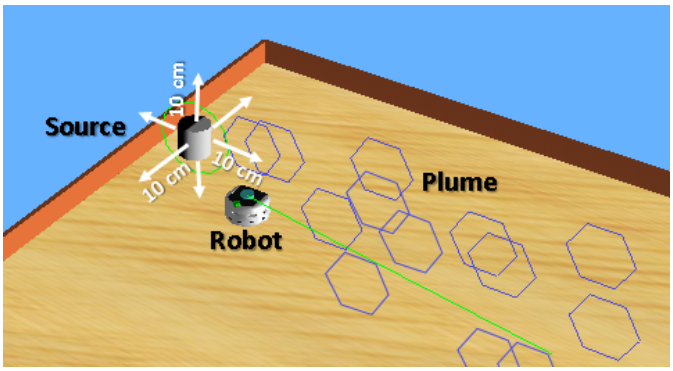


Figure 17.2 – Margin of the source area in simulation.

17.2.2 Performance evaluation

In order to assess the performance of the proposed method, a comparison with a similar algorithm of the same category would be necessary. Since the contribution of this work is the crosswind local search in 3D, we compare its performance with its 2D counterpart.

Baseline 2D method

We have chosen the Surge-Cast algorithm [16] as reference in terms of performances, as it performs a crosswind local search, similar to the proposed method but in 2D. We have adapted this algorithm to the procedure of our 3D proposed method in order to have a fair comparison.

Assuming that the X-axis is towards the upwind direction, the strategy of this reference method would be the following: after establishing the gas concentration threshold at the fixed altitude, the robot moves to a random position on the Y-axis, at the beginning of the experimental environment. At this point, it performs LW on the Y-axis (crosswind) in order to find the plume. Therefore, for this method, due to the restriction on the Z-axis, the coordinates of a new step taken from the position (x_0, y_0, z_0) would be as expressed in Eq. (17.4).

$$\begin{aligned} x &= x_0 \\ y &= y_0 + M_l \,.\, \sin(B_{Ta}) \\ z &= z_0 \end{aligned} \tag{17.4}$$

with $B_{Ta} = 0$ if $T_a < \pi$, and $B_{Ta} = 1$ otherwise.

Once the plume is retrieved, the robot moves upwind towards the X-axis, while it remains in the plume. As soon as the plume is lost, the robot moves crosswind for a distance of D_{cast} (here 43 cm based on [33]). If the plume is found during this movement, the upwind surge is resumed, otherwise the robot performs another crosswind surge, in the opposite direction, for a distance of $2 \times D_{cast}$ (see Figure 17.1d). This local search continues by exploring more either sides each time, until the plume is reacquired. Finally, the algorithms ends either when the

Table 17.1 – Setup parameters used in the experiments and simulations.

Setup	Wind speed	Release rate	Concentration threshold
A	0.2 m/s	low	70%
B	0.2 m/s	low	90%
C	0.2 m/s	high	70%
D	0.2 m/s	high	90%
E	0.9 m/s	low	70%
F	0.9 m/s	low	90%
G	0.9 m/s	high	70%
H	0.9 m/s	high	90%

source is located, which makes the run successful, or the time limit is reached which means a failure.

Important parameters

The performances of our method and the reference algorithm have been verified in different setups involving different parameters. In the present section, we describe the parameters that we varied as well as their potential impact on the results. The chosen values for each of the parameters are reported in Table 17.1.

- *Wind speed and source release rate* - As mentioned earlier, wind speed and source release rate of the source have a high impact on the shape of the plume. The release rate controls the its richness in terms of numbers of gas patch and concentrations. The wind speed affects that width of the plume.
- *Gas concentration threshold* - Through the introduction of a gas concentration threshold, the robot perceives the plume in a "binary" form. In particular, the concentration threshold is the value beyond which the robot is considered being inside the plume, and outside otherwise. Therefore, if this value is chosen too high, the plume seen by the robot is very narrow and patchy. On the other hand, a very low value yields a misleadingly large plume perceived by the robot.

  In the simulations and physical experiments of our previous contribution [95] we set the threshold empirically to 0.7 and 0.9. These values mean that, respectively, the highest 70% and 90% of gas concentration measurements at the initial phase are considered inside the plume. The corresponding to the gas concentration might change between the runs of a single setup.
- *Initial position of the robots with respect to the source* - The position where the robot starts the operation has a high impact on the performance of the algorithms. In order to be fair between the runs of both algorithms, we chose to place the robot on a random initial position on the crosswind section, and 10 m far from the source in the X-axis

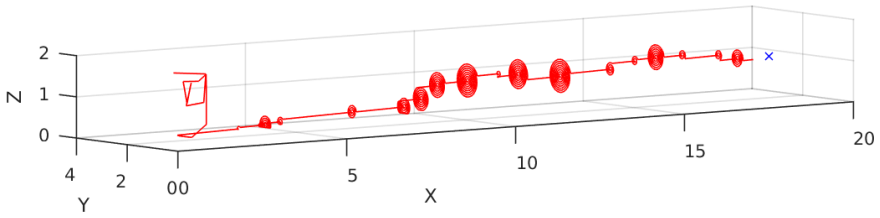


Figure 17.3 – Trajectory of the proposed 3D algorithm in simulation.

(upwind direction). As the 2D algorithm needs to be restrained on a fixed altitude, we have chosen to set it at a certain altitude, more precisely 10 cm below the height of the source (in our previous work [95] set at $z = 50$ cm).

Metrics

The performances of both algorithms have been previously evaluated using two metrics: success rate and execution time.

A run is considered successful if the robot reaches the source area while being inside the plume, within the time window of 12 min (arbitrary value, twice longer than most of usual successful runs). Therefore, for each setup, the number of successful runs over the total number of runs is called success rate s_r. The other metric would simply be the time spent by the robot in the experiment. It is calculated from the moment when it starts the search from its initial position, until it declares success by reaching the source. For failed runs, this metric is not taken into account. As the robot travels with a constant speed of 50 mm/s, and no turnings are required in the movements, it is fair to simply consider the time instead of the traveled distance.

Simulation experiments

Before evaluating the proposed 3D bio-inspired algorithm in physical experiments, we have developed and tested it in simulation leveraging, as before, Webots.

A trajectory of the robot is shown in Figure 17.3. For different setups equivalent of the ones detailed in Table 17.1, the method has been run 20 times and the results are summarized in Figure17.4. As expected, the performance of the algorithm is outstanding even in complex environmental conditions (e.g., setup A). In lower wind-speed the general performance in terms of success rate is not affected, however in terms of execution time, setups with lower wind speeds spent about 100 s more for a given gas concentration threshold and source release rate. This is due to the time the robot has to spend on local search when it loses the plume. The two other parameters, i.e. source release rate and threshold, did not have a significant impact on the performance. This implies that the algorithm is robust to these parameters.

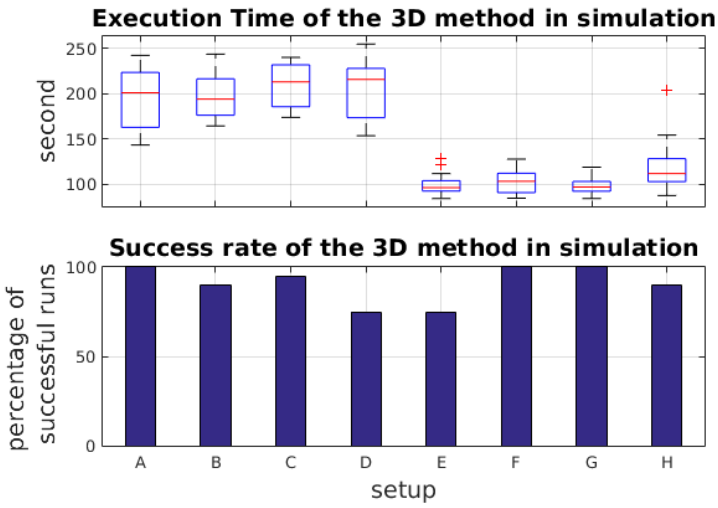


Figure 17.4 – Performances of the proposed 3D algorithm in simulation in different setups.

Physical experiments

In order to assess the performance of the algorithms in a repeatable fashion, as in the previous chapters, physical experiments were carried out in the wind tunnel. We mounted a wheeled Khepera IV robot on the traversing system (see Figure 17.5). The system composed of the traversing system and the Khepera robot represents an emulated flying robot, able to move on three axes. Thus, given a position $p(x, y, z)$, the flying robot moves towards p with a constant speed.

For this work, we have carried out experiments with 8% (low) and 18% (high) of the pump power as gas source, using acetone instead of ethanol as liquid to be vaporized. In both pump regimes, a human nose was not able to detect the acetone smell in the environment.

Figure 17.6 shows two trajectories performed using the two evaluated algorithms. Both trajectories start with a crosswind search, on the Y-Z plane for the 3D algorithm and on the Y-axis for the 2D method. In the rest of the trajectories, upwind surge and local search are used consequentially. It should be noted that the movement of the traversing system is limited to its working area borders, therefore some trajectories, such as the spiralling, are truncated at the margins of this area. The blue cross shows the position of the source.

We have evaluated the presented method 10 times for each setup exposed in Tab. 17.1, and the results are shown in Figure 17.7.

The 3D method shows, as expected, outstanding performance in case of high-speed wind and high release rate, since the amount of information transmitted from the source to the robot is sufficient to guide it correctly.

It is interesting to note that, in the setup F (i.e., wind speed = 0.9 m/s, source rate = low and

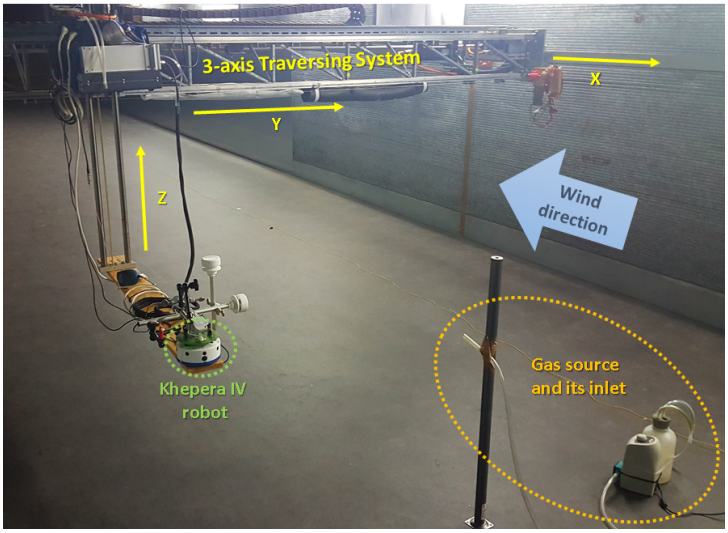


Figure 17.5 – Wind tunnel equipped with a 3-axis traversing system, the Khepera IV robot and an gas source.

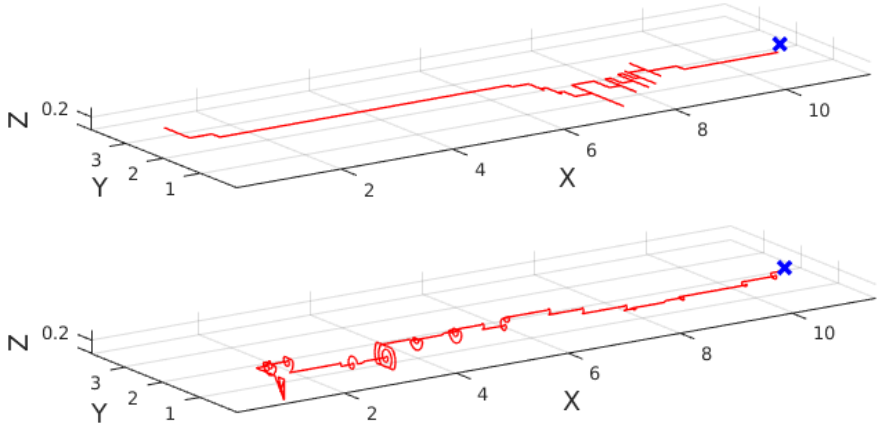


Figure 17.6 – Trajectory of two successful runs of both algorithms (2D on top and 3D on bottom) in the wind tunnel, with the wind speed of 0.9 m/s, source release rate of 18%, and concentration threshold of 0.9. The blue cross shows the position of the source.

concentration threshold = 90%), even though the source release rate is not high, the higher gas concentration threshold, compared to the case E, compensated for the environmental conditions and the 3D algorithm remained totally successful. However, this happens for the case B and D, where the wind speed was very low. This proves that the wind speed is more significant than the other two parameters and thus the baseline proposed method, while being generally successful in different environmental conditions, is sensitive to the cases characterized by a low wind speed.

The same outcome can be seen in Figure 17.8 that summarizes the results of the 2D baseline method for sets of five runs for each setup. The group of results corresponding to conditions with high wind speed have better performance in terms of success rate and spent time. As opposed to the 3D method, the 2D algorithm does not yield satisfactory results with low wind speed, and is less efficient is general.

Since the plume gets wider (in crosswind plane) as it travels far from the source, the robot can sense the plume at the beginning of the experiment even if it is located at a lower altitude with respect to the source. However, by tracking the plume towards the upwind direction and getting closer to the source where the plume is narrower, it becomes harder to sense the plume and thus the robot has to perform multiple local search in the process. This is why a 2D algorithm results in not a reliable performance, even in case the source is located only 10 cm higher than the operation plane of the robot.

17.2.3 Conclusion

The 3D bio-inspired algorithm was validated in simulation, evaluated in the wind tunnel, and benchmarked with a well-established 2D bio-inspired algorithm under different environmental conditions. As other bio-inspired algorithms, it needs very little a priori knowledge and computational resources. Our conclusion is that unless the source is located on the ground, an aerial vehicle and a 3D algorithm are needed to robustly find the source. Studying three main parameters, the experiments showed that the wind speed is more significant than the other two parameters and thus the proposed method, while being generally successful in different environmental conditions, is sensitive to situations characterized by low wind speed.

This method was later deployed on a flying vehicle and presented in [103].

17.3 Performance Comparison with the STE Algorithm

In this section, we will use the previously presented 3D bio-inspired algorithm as benchmark for the 3D STE algorithm. Since the benchmark algorithm covers the plume acquisition sub-task as well, the position of the robot is randomized on the Y-Z plane in both algorithms.

We evaluated both methods under the different environmental conditions reported in Table 8.1. The results depicted in Figure 17.9 show that the traveled distance is affected by the wind speed

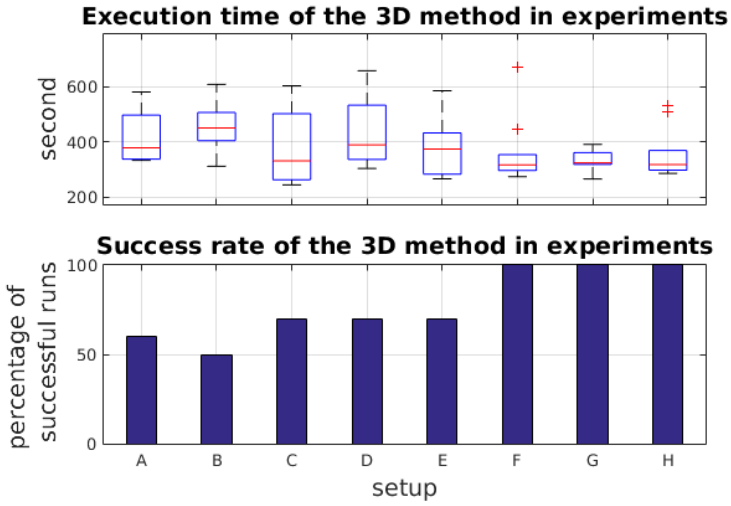


Figure 17.7 – Performances of proposed 3D algorithm in physical experiments in different setups.

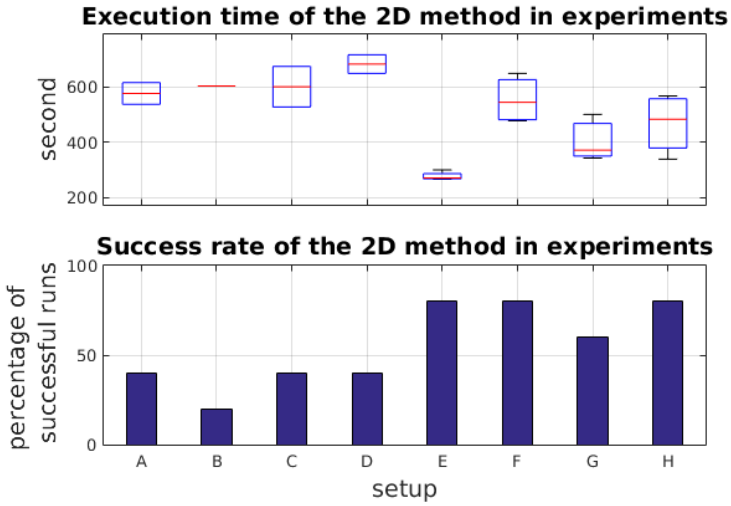


Figure 17.8 – Performances of Surge-Cast algorithm in physical experiments in different setups.

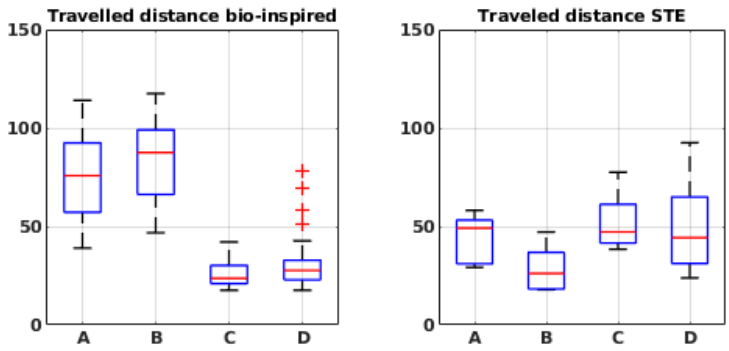


Figure 17.9 – Performance comparison between our 3D bio-inspired algorithm and STE.

for the bio-inspired algorithm, which is not the case for STE. Indeed, the bio-inspired algorithm shows a better performance in high wind speed compared to STE, but it lacks robustness in low wind speed. STE, however, shows a consistent performance in all environmental conditions.

18 Conclusion

We have compared two variants of our proposed STE algorithm with two competitive bio-inspired algorithms.

In the 2D search space, the statistical results on traveled distance show similar performances for both algorithms in different environmental conditions. However, the STE algorithm, being able to tackle the GSL problem in all its sub-tasks, shows a consistent performance even when the starting position of the robot is placed outside of the plume.

In the 3D search space, according to the results, even though the bio-inspired method performs with a lower traveled distance in high wind speed, the STE algorithm shows again superior consistency and robustness to the evaluated environmental conditions.

Additionally, the STE algorithm presents features that make it a better algorithm overall : it addresses all three stages of the GSL problem, it provides a richer set of information about the source characteristics other than its position, it provides the estimation uncertainty on the source location, an insightful feature to show the reliability of the results.

Conclusion Part VI

19 Conclusion

Over the course of this thesis, we studied the problem of GSL through the formal framework of STE. Our proposed STE algorithm estimates the parameters of a plume model using the Bayesian approach, and navigates the robot through the environment using a POMDP strategy that rewards high information gain. The cycle between estimation and navigation allows for localizing the source of a gas release with high accuracy. On top of the source location, the algorithm provides, as output, other relevant information on the source and the plume, as well as an uncertainty level that shows the confidence level on the output data. The latter is related to the probabilistic nature of the method, which brings other advantages, such as flexibility to the type of sensing asset. The drawback, being the high computational cost can be reduced with probabilistic sampling techniques, and in any case, appears to be insignificant nowadays with the fast evolution of powerful pocket-size computation units.

To demonstrate the versatility of the proposed algorithm, and with the intention of proposing solutions to realistic scenarios, the algorithm was further enriched with new features that allowed for adaptation to scenarios with different aspects of complexities. Three main axes of complexity were considered for the scenarios, namely (i) spatial dimension, (ii) sensing system, and (iii) environment complexity. Our algorithm was first adapted to the underlying scenario and then thoroughly evaluated in it. When applicable, the performances were also compared with the core method to show the effectiveness of the adaptation. The results showed that with the right coordination strategy, the efficiency of the algorithm increases when deployed on a distributed system. Additionally, the underlying plume model has a major impact on the type of environment it can be applied to, but with the suitable model, it can be applied to complex environment. However, designing a suitable model for any type of environment might be non-trivial. Therefore, we believe that the application in real scenarios is currently limited by to the lack of a suitable model.

Our proposed algorithm was evaluated in each setup first in a high-fidelity simulation, and then, if possible, in a wind tunnel leveraging one or more robotic assets, allowing for repeatable and reliable experimentation. Additionally, the performance of our method was benchmarked with competitive bio-inspired algorithms, showing not only quantitative competitiveness but

also superior robustness to a variety of environmental conditions.

19.1 Summary of Contributions

This thesis presents various contributions in the field of GSL through the STE algorithm as well as its adaptations to complex scenarios. A summary of the contributions of this work are presented in the following.

- We propose a STE algorithm with a novel navigation technique that leads the robot toward informative points to accelerate the localization of a gas source. After systematic evaluations that allowed for finding the best values for algorithmic values, this algorithm was thoroughly evaluated in simulation and in physical experiments in a wind tunnel with different environmental conditions. Two main environmental parameters were considered, namely the wind speed and the source release rate. The algorithm localized the source with high accuracy and showed robustness to the evaluated environmental conditions.
- We extended the applicability of our method to a 3D search space, by replacing the underlying plume model with its 3D variant. It was then evaluated in simulation in the same environmental conditions as the 2D setup, and showed successful in all of them.
- With the intention of reducing the amount of a priori data that the algorithm requires from the environment, we adapted our algorithm to operate in an unknown environment, where no global localization is available and the map of the environment is not provided to the robot. Therefore, the robot showed ability to localize the source with odometry and local maps, while substantially decreasing the traveled distance. However, a systematic evaluation of this algorithmic variant in simulation and in physical reality showed lack of robustness to diverse environmental conditions.
- To leverage the versatility of the STE framework, we deployed our algorithms on a homogeneous multi-robot system. We designed three coordination strategies with increasing levels of coordination among robots. The system was evaluated with up to three robots, both in simulation and in physical experiments. Additionally, an enhanced navigation technique was introduced that allowed for further reducing the traveled distance. The highest level of coordination, where the robots were able to (i) coordinate their source declaration, (ii) share their samples, and (iii) coordinate their movements with each other, showed to be the most efficient strategy in terms of traveled distance and experiment duration.
- To extend the scope of the proposed algorithm to cluttered environments, we successfully developed a novel data-driven plume model and integrated it in our STE algorithm. The model was deduced from feeding in training data, i.e. simulated concentration maps in randomly designed cluttered environments, to a convolutional neural network.

The model was first evaluated with nine arbitrarily designed maps and, after showing success, it was integrated in our STE algorithm. The algorithm was then evaluated in simulation with randomized source positions in cluttered environments and showed great performance.

- The core STE method was benchmarked with two competitive bio-inspired algorithms, in a 2D and a 3D search spaces. The benchmark algorithms themselves were already benchmarked with well-established moth-inspired algorithms and showed great success rate in comparison. Our STE algorithm, while being similar is terms of time and energy efficiency to the benchmark algorithms, presented additional advantages over bio-inspired algorithms, namely, covering all three sub-tasks of the GSL problem with a single method, providing a richer set of information about the source, and achieving superior robustness to diverse environmental conditions.

To highlight the novelty of this thesis over the state of the art, here we briefly mention the differences with the most prominent and recent related works.

The estimation part of our method is mostly inspired by the STE literature (e.g., [41, 62]). Since the STE algorithm was traditionally used for off-line applications, the estimation part was already well-studied. However, navigation strategies were rarely coupled with STE for closed-loop applications. Additionally, most of the existing works in the literature were done in simulation only (e.g., [65, 66, 68]). Therefore, to be best of our knowledge, our STE method is one of the very few that was deployed on real mobile robots (e.g., [55]), and among those few, our navigation algorithm is the only one designed natively based on probabilistic reasoning. Finally, the systematic experimentations that we carried out in a controlled physical environment were also missing in the literature.

[46] is a thesis with similarities in terms of general probabilistic approach to the GSL problem applied to a multi-robot system. Although the STE algorithm is not mentioned, the parameters of a gas dispersion model is probabilistically computed based on the robots' observations, similarly to our method. The main difference in the estimation part of the two approches is the choice of the plume model. [46] uses advection-diffusion partial differential equations to model gas dispersion, which is a richer and more accurate model than our pseudo-gaussian plume model. However, in Chapter 14 we studied a data-driven plume model than showed to have the potential of becoming a suitable surrogate model. The exploration technique of [46] guides the robots toward points with higher uncertainties. While in our method, to balance the exploration and exploitation, we also leverage the most likely position of the source to accelerate the search. Moreover, the multi-robot implementations are different but they are both distributed and the results are unfortunately hard to compare due to major differences in experimentation setups.

19.2 Discussion and outlook

We believe that, in the near future, GSL algorithms will be deployed on autonomous robots and be helpful in localizing sources of gaseous releases in different applications, which could take place on Earth or even on further planets such as Mars.

This thesis presents a rich algorithm, with high potential for becoming a formal solution to the complex problem of GSL. Different challenges of realistic environments were successfully addressed with our proposed method by adding new features. However, for technology to reach the applicability of such methods in real environments, more steps need to be taken and more challenges need to be overcome. In this section, we discuss the limits of our current implementation and we present our outlook on this work.

The core algorithm consists of a combination of many methods that can be replaced by other methods to improve the efficiency of the framework. The main parts of the algorithm, being estimation and navigation, are interconnected, and therefore, an improvement on one, brings improvement on the entire algorithm. For instance, the estimation is done probabilistically leveraging a Bayesian approach, but an optimization algorithm could be more time-efficient in comparison, and therefore preferable. Moreover, the range of the sought parameters of the source were chosen arbitrarily, while they could be set dynamically without a static range. This would improve the robustness of the method in environmental conditions. Finally, the hyper-parameters of the method could be optimized using a global optimization algorithm.

The method was adapted to a 3D search space, but unfortunately, it was only evaluated in simulation. To validate its performance in a real environment, it should be run on a flying vehicle. However, the impact of the propellers will certainly affect the gas concentration measurements, which means that it would be necessary to consider a larger measurement error in the parametrization of the algorithm. On the other hand, it might be best to modify the navigation strategy to not move toward the source, which would prevent mixing high concentration gas patches in the air to the point of saturation. Further investigation on this problem would be necessary before a real deployment.

In case of an unknown environment, our method showed promising in terms of traveled distance, but not in terms of robustness to environmental conditions. We suggest a combination of global estimation, with dynamic parameter range, and local navigation. This combined method would allow the robot to travel less distance, but to estimate the source position at a global level.

To extend the deployment flexibility of the distributed variant of our STE algorithm, we can include static sensors to the system to further increase the power efficiency and sensing accuracy. The system can also include flying vehicles, coupled with ground robots or static sensors, to cover a 3D search space in a more energy-efficient way. Many parameters of such a system must be studied as a function of the environment, such as the number of sensing assets, their positioning, their mobility, and above all, the coordination strategy among them.

The coordination strategy is a non-trivial task in a heterogeneous system, which should be thoroughly studied. It should be designed in order to minimize the redundancy between the roles of different sensing assets in the performance of the system.

For the proposed algorithm to be applicable in realistic indoor environments, a data-driven plume model was developed that takes as input only the coordinates of the source location. Whereas for the model to be applicable in real scenarios, it should also include other source characteristics, such as the source release rate and the wind speed. Adding these two variables to the model requires as least three times more training data, and might involve a more complex neural network. Moreover, after integrating the model to the STE framework, we only evaluated it in simulation. Physical experiments would be necessary before claiming its applicability in real environment. Finally, due to its high computational power requirements, the model cannot run on a Khepera IV robot in real time, therefore it needs to run off board, unless a more powerful computation unit is used on an autonomous robot. Further optimization on the neural network and the spatial resolution of the training maps could be done to reduce the complexity of the model and hence its computational cost.

Finally, each axis of complexity was tackled individually in this thesis. Combining one or more scenario will introduce new challenges and would improve the applicability of the algorithm in real deployments.

Bibliography

[1] G. Etiope and D. Z. Oehler, "Methane spikes, background seasonality and non-detections on Mars: A geological perspective," *Planetary and Space Science*, vol. 168, pp. 52–61, 2019.

[2] J. A. Farrell, J. Murlis, X. Long, W. Li, and R. T. Cardé, "Filament-Based Atmospheric Dispersion Model to Achieve Short Time-Scale Structure of Odor Plumes," *Environmental Fluid Mechanics*, vol. 2, no. 1-2, pp. 143–169, 2002.

[3] B. L. Villarreal, G. Olague, and J. L. Gordillo, "Synthesis of odor tracking algorithms with genetic programming," *Neurocomputing*, vol. 175, pp. 1019–1032, 2016.

[4] C. Li, Z. Yang, G. Cui, and B. Jin, "Odor Source Localization Research of Mobile Robots in Indoor Environments," *Applied Mechanics and Materials*, vol. 441, pp. 796–800, 2013.

[5] J. Murlis, J. S. Elkinton, and R. T. Cardé, "Odor Plumes And How Insects Use Them," *Annual Review of Entomology*, vol. 37, no. 1, pp. 505–532, 1992.

[6] Y. Zhang, X. Ma, Y. Miao, Z. Yuli, M. A. Xiaoping, and M. Yanzi, "Localization of multiple odor sources using modified glowworm swarm optimization with collective robots," in *Proceedings of the 30th Chinese Control Conference*, pp. 1899–1904, 2011.

[7] P. P. Neumann, *Gas Source Localization and Gas Distribution Mapping with a Micro-Drone*. PhD thesis, Freien Universität Berlin, 2013.

[8] A. Hayes, A. Martinoli, and R. M. Goodman, "Distributed Odor Source Localization," *IEEE Sensors Journal*, vol. 2, no. 3, pp. 260–271, 2002.

[9] Z. Pasternak, F. Bartumeus, and F. W. Grasso, "Lévy-taxis: a novel search strategy for finding odor plumes in turbulent flow-dominated environments," *Journal of Physics A: Mathematical and Theoretical*, vol. 42, no. 43, p. 434010, 2009.

[10] A. Marjovi and L. Marques, "Optimal swarm formation for odor plume finding," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2302–2315, 2014.

[11] R. Rozas, J. Morales, and D. Vega, "Artificial smell detection for robotic navigation," in *Fifth International Conference on Advanced Robotics 'Robots in Unstructured Environments*, pp. 1730–1733 vol.2, 1991.

[12] V. Genovese, P. Dario, R. Magni, L. Odetti, and S. Anna, "Self-organizing behavior and swarm intelligence in a pack of mobile miniature robots in search of pollutants," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 243–248, 1992.

[13] G. Kowadlo and R. A. Russell, "Robot Odor Localization: A Taxonomy and Survey," *The International Journal of Robotics Research*, vol. 27, no. 8, pp. 869–894, 2008.

[14] H. Ishida, K. Hayashi, M. Takakusaki, T. Nakamoto, T. Moriizumi, and R. Kanzaki, "Odour-source localization system mimicking behaviour of silkworm moth," *Sensors and Actuators*, pp. 225–230, 1996.

[15] T. Lochmatter and A. Martinoli, "Simulation experiments with bio-inspired algorithms for odor source localization in laminar wind flow," in *Machine Learning and Applications*, pp. 437–443, 2008.

[16] T. Lochmatter and A. Martinoli, "Tracking Odor Plumes in a Laminar Wind Field with Bio-inspired Algorithms," in *11th Int. Symposium on Experimental Robotics*, vol. 54, pp. 473–482, 2008.

[17] T. Lochmatter and A. Martinoli, "Understanding the Potential Impact of Multiple Robots in Odor Source Localization," in *Distributed Autonomous Robotic Systems*, pp. 239–250, 2008.

[18] T. Lochmatter, N. Heiniger, and A. Martinoli, "Localizing an odor source and avoiding obstacles: experiments in a wind tunnel using real robots," in *13th International Symposium on Olfaction and Electronic Nose*, vol. 1137, pp. 69–72, 2009.

[19] V. H. Bennetts, A. J. Lilienthal, P. Patrick, and M. Trincavelli, "Mobile robots for localizing gas emission sources on landfill sites: is bio-inspiration the way to go?," *Bioinspired solutions to the challenges of chemical sensing*, vol. 4, p. 164, 2012.

[20] V. Gazi, L. Marques, and R. Ordonez, "Robot Swarms : Dynamics and Control," *Mobile Robots for Dynamic Environments. ASME*, pp. 81–85, 2015.

[21] L. Marques and U. Nunes, "Particle swarm-based olfactory guided search," *Autonomous Robots*, vol. 20, no. 3, pp. 277–287, 2006.

[22] W. Jatmiko, K. Sekiyama, and T. Fukuda, "A PSO-Based Bobile Robot for Odor Source Localization in Dynamic Advection-Diffusion with Obstacles Environment: Theory, Simulation and Measurement," *IEEE Computational Intelligence Magazine*, vol. 2, no. 2, pp. 37–51, 2007.

[23] D. W. Gong, C. L. Qi, Y. Zhang, and M. Li, "Modified particle swarm optimization for odor source localization of multi-robot," *IEEE Congress of Evolutionary Computation*, pp. 130–136, 2011.

[24] J. Zhang, D. Gong, and Y. Zhang, "A niching PSO-based multi-robot cooperation method for localizing odor sources," *Neurocomputing*, vol. 123, pp. 308–317, 2014.

[25] N. Li, Q. Lu, Y. He, and J. Wang, "Particle Swarm Optimization Based on Shannon's Entropy for Odor Source Localization," in *International Conference on Life System Modeling and Simulation and International Conference on Intelligent Computing for Sustainable Energy and Environment*, pp. 140–148, 2014.

[26] K. Krishnanand and D. Ghose, "A glowworm swarm optimization based multi-robot system for signal source localization," in *Design and control of intelligent robotic systems*, pp. 49–68, Springer, 2009.

[27] Y. Kuwana, I. Shimoyama, Y. Sayama, and H. Miura, "Synthesis of Pheromone-Oriented Emergent Behavior of a Silkworm Moth," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 1722–1729, 1996.

[28] R. D. Beer and J. C. Gallagher, "Evolving Dynamical Neural Networks for Adaptive Behavior," *Adaptive behavior*, vol. 1, pp. 91–122, 1992.

[29] G. C. H. E. de Croon, L. M. O'Connor, C. Nicol, and D. Izzo, "Evolutionary robotics approach to odor source localization," *Neurocomputing*, vol. 121, no. 1, pp. 481–497, 2013.

[30] T. Lochmatter, E. Aydin Göl, I. Navarro, and A. Martinoli, "A Plume Tracking Algorithm Based on Crosswind Formations," in *Distributed Autonomous Robotic Systems*, pp. 91–102, 2010.

[31] J. Soares, *Formation-Based Odour Source Localisation Using Distributed Terrestrial and Marine Robotic Systems*. Phd thesis no. 7080, EPFL, 2016.

[32] J. M. Soares, A. Marjovi, J. Giezendanner, A. Kodiyan, A. P. Aguiar, M. Pascoal, and A. Martinoli, "Towards 3-D Distributed Odor Source Localization : An Extended Graph-Based Formation Control Algorithm for Plume Tracking," in *IEEE International Conference on Intelligent Robots and Systems*, pp. 1729–1736, 2016.

[33] T. Lochmatter, *Bio-inspired and probabilistic algorithms for distributed odor source localization using mobile robots*. Phd thesis no. 4628, EPFL, 2010.

[34] M. Vergassola, E. Villermaux, and B. I. Shraiman, "' Infotaxis ' as a strategy for searching without gradients," *Nature*, vol. 445, pp. 406–409, 2007.

[35] N. Voges, A. Chaffiol, P. Lucas, and D. Martinez, "Reactive Searching and Infotaxis in Odor Source Localization," *PLoS Computational Biology*, vol. 10, no. 10, 2014.

[36] J. D. Rodriguez, D. Gomez-Ullate, and C. Mejia-Monasterio, "Limits on the performance of Infotaxis under inaccurate modelling of the environment," *arXiv preprint arXiv:1408.1873*, 2014.

[37] S. Pang and J. A. Farrell, "Chemical Plume Source Localization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 36, no. 5, pp. 1068–1080, 2006.

[38] J.-G. Li, Q.-H. Meng, Y. Wang, and M. Zeng, "Odor source localization using a mobile robot in outdoor airflow environments with a particle filter algorithm," *Autonomous Robots*, vol. 30, no. 3, pp. 281–292, 2011.

[39] M. Hutchinson, H. Oh, and W. H. Chen, "A review of source term estimation methods for atmospheric dispersion events using static or mobile sensors," *Information Fusion*, vol. 36, pp. 130–148, 2017.

[40] B. Ristic, M. Morelande, and A. Gunatilaka, "Information driven search for point sources of gamma radiation," *Signal Processing*, vol. 90, no. 4, pp. 1225–1239, 2010.

[41] A. Keats, E. Yee, and F. S. Lien, "Bayesian inference for source determination with applications to a complex urban environment," *Atmospheric Environment*, vol. 41, no. 3, pp. 465–479, 2007.

[42] J. A. Farrell, S. Pang, and W. Li, "Plume Mapping via Hidden Markov Methods," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 33, no. 6, pp. 850–863, 2003.

[43] J. L. Blanco, J. G. Monroy, J. Gonzalez-Jimenez, and A. J. Lilienthal, "A Kalman Filter Based Approach to Probabilistic Gas Distribution Mapping," in *SAC '13 Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pp. 217–222, 2013.

[44] A. J. Lilienthal, M. Reggente, M. Trincavelli, J. L. Blanco, and J. Gonzalez, "A Statistical Approach to Gas Distribution Modelling with Mobile Robots – The Kernel DM + V Algorithm," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 570–576, 2009.

[45] A. Marjovi and L. Marques, "Multi-robot odor distribution mapping in realistic time-variant conditions," in *IEEE International Conference on Robotics and Automation*, pp. 3720–3727, IEEE, 2014.

[46] T. Wiedemann, *Domain Knowledge Assisted Robotic Exploration and Source Localization*. PhD thesis, Örebro University, School of Science and Technology, 2020.

[47] M. Reggente and A. J. Lilienthal, "The 3D-Kernel DM+V/W Algorithm: Using Wind Information in Three Dimensional Gas Distribution Modelling with a Mobile Robot," in *IEEE Sensors*, pp. 999–1004, 2010.

[48] G. Cabrita and L. Marques, "Divergence-based odor source declaration," in *9th Asian Control Conf.*, pp. 1–6, 2013.

[49] G. Kowadlo, D. Rawlinson, R. A. Russell, and R. Jarvis, "Bi-modal search using complementary sensing (olfaction/vision) for odour source localisation," in *IEEE Int. Conf. on Robotics and Automation*, pp. 2041–2046, 2006.

[50] SGX Sensortech technologies, "MiCS-5521 CO/VOC sensor."

[51] T. Lochmatter, P. Roduit, C. Cianci, N. Correll, J. Jacot, and A. Martinoli, "Swistrack - A flexible open source tracking software for multi-agent systems," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 4004–4010, 2008.

[52] O. Michel, "Webots TM : Professional Mobile Robot Simulation," *International Journal of Advanced Robotic Systems*, vol. 1, no. 1, pp. 39–42, 2004.

[53] Wikibooks, "Webots odor simulation — wikibooks, the free textbook project," 2010. [Online; accessed 7-November-2016].

[54] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm Intelligence*, vol. 1, no. 1, pp. 33–57, 2007.

[55] J. R. Bourne, E. R. Pardyjak, and K. K. Leang, "Coordinated Bayesian-Based Bioinspired Plume Source Term Estimation and Source Seeking for Mobile Robots," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 967–986, 2019.

[56] K. J. Long, S. E. Haupt, and G. S. Young, "Assessing sensitivity of source term estimation," *Atmospheric Environment*, vol. 44, no. 12, pp. 1558–1567, 2010.

[57] B. Addepalli, K. Sikorski, E. Pardyjak, and M. Zhdanov, "Source characterization of atmospheric releases using stochastic search and regularized gradient optimization," *Inverse Problems in Science and Engineering*, vol. 19, no. 8, pp. 1097–1124, 2011.

[58] M. Newman, K. Hatfield, J. Hayworth, P. Rao, and T. Stauffer, "A hybrid method for inverse characterization of subsurface contaminant flux," *Journal of Contaminant Hydrology*, vol. 81, no. 1-4, pp. 34–62, 2005.

[59] L. C. Thomson, B. Hirst, G. Gibson, S. Gillespie, P. Jonathan, K. D. Skeldon, and M. J. Padgett, "An improved algorithm for locating a gas source using inverse methods," *Atmospheric Environment*, vol. 41, no. 6, pp. 1128–1134, 2007.

[60] C. T. Allen, S. E. Haupt, and G. S. Young, "Source characterization with a genetic algorithm–coupled dispersion–backward model incorporating scipuff," *Journal of Applied Meteorology and Climatology*, vol. 46, no. 3, pp. 273–287, 2007.

[61] S. E. Haupt, G. S. Young, and C. T. Allen, "A genetic algorithm method to assimilate sensor data for a toxic contaminant release," *Journal of Computers*, vol. 2, no. 6, pp. 85–93, 2007.

[62] G. Johannesson, B. Hanley, and J. Nitao, "Dynamic Bayesian models via Monte Carlo-an introduction with examples," tech. rep., Lawrence Livermore National Lab., Livermore, CA (US), 2004.

[63] A. Gunatilaka, B. Ristic, A. Skvortsov, and M. Morelande, "Parameter estimation of a continuous chemical plume source," in *2008 11th International Conference on Information Fusion*, pp. 1–8, IEEE, 2008.

[64] A. Wawrzynczak, P. Kopka, and M. Borysiewicz, "Sequential Monte Carlo in Bayesian assessment of contaminant source localization based on the sensors concentration measurements," in *International conference on parallel processing and applied mathematics*, pp. 407–417, Springer, 2013.

[65] Y. Kuroki, G. S. Young, and S. E. Haupt, "UAV navigation by an expert system for contaminant mapping with a genetic algorithm," *Expert Systems with Applications*, vol. 37, no. 6, pp. 4687–4697, 2010.

[66] B. Hirst, P. Jonathan, F. G. del Cueto, D. Randell, and O. Kosut, "Locating and quantifying gas emission sources using remotely obtained concentration data," *Atmospheric Environment*, vol. 74, pp. 141–158, 2013.

[67] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[68] B. Ristic and A. Gunatilaka, "Information driven localisation of a radiological point source," *Information Fusion*, vol. 9, no. 2, pp. 317–326, 2008.

[69] R. Madankan, P. Singla, and T. Singh, "Optimal information collection for source parameter estimation of atmospheric release phenomenon," *Proceedings of the American Control Conference*, pp. 604–609, 2014.

[70] S. P. Arya, *Air pollution meteorology and dispersion*. Oxford University Press, 1999.

[71] W. R. Gilks, S. Richardson, and D. Spiegelhalter, *Markov chain Monte Carlo in practice*. CRC press, 1995.

[72] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[73] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[74] H. Ishida, K. Yoshikawa, and T. Moriizumi, "Three-Dimensional Gas-Plume Tracking Using Gas Sensors and Ultrasonic Anemometer," in *IEEE Sensors*, pp. 1175–1178, 2004.

[75] R. A. Russell, "Tracking Chemical Plumes in 3-Dimensions," in *International Conference on Robotics and Biomimetics*, pp. 31–36, 2006.

[76] B. Gao, H. Li, W. Li, and F. Sun, "3D Moth-inspired chemical plume tracking and adaptive step control strategy," *Adaptive Behavior*, vol. 24, no. 1, pp. 52–65, 2016.

[77] V. Braitenberg, *Vehicles: Experiments in synthetic psychology*. MIT press, 1986.

[78] J. Ruddick, A. Marjovi, F. Rahbar, and A. Martinoli, "Design and Performance Evaluation of an Infotaxis-Based Three-Dimensional Algorithm for Odor Source Localization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1413–1420, 2018.

[79] M. Park and H. Oh, "Cooperative information-driven source search and estimation for multiple agents," *Information Fusion*, vol. 54, pp. 72–84, 2020.

[80] F. Rahbar, A. Marjovi, and A. Martinoli, "An algorithm for odor source localization based on source term estimation," in *IEEE International Conference on Robotics and Automation*, pp. 973–979, 2019.

[81] R. Sykes, S. Parker, D. Henn, and R. Gabruk, "Scipuff—a generalized dispersion model," in *Air Pollution Modeling and Its Application XI*, pp. 425–432, Springer, 1996.

[82] M. D. Ribeiro, A. Rehman, S. Ahmed, and A. Dengel, "DeepCFD: Efficient Steady-State Laminar Flow Approximation with Deep Convolutional Neural Networks," 2020. arXiv:2004.08826v2 [physics.comp-ph].

[83] X. Guo, W. Li, and F. Iorio, "Convolutional neural networks for steady flow approximation," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, vol. 13-17-Augu, pp. 481–490, 2016.

[84] D. Stoecklein, K. G. Lore, M. Davies, S. Sarkar, and B. Ganapathysubramanian, "Deep Learning for Flow Sculpting: Insights into Efficient Learning using Scientific Simulation Data," *Scientific Reports*, vol. 7, no. March, pp. 1–11, 2017.

[85] T. Georgiou, S. Schmitt, M. Olhofer, Y. Liu, T. Bäck, and M. Lew, "Learning fluid flows," in *International Joint Conference on Neural Networks*, pp. 1–8, 2018.

[86] B. Kim, V. C. Azevedo, N. Thuerey, T. Kim, M. Gross, and B. Solenthaler, "Deep Fluids: A Generative Network for Parameterized Fluid Simulations," *Computer Graphics Forum*, vol. 38, no. 2, pp. 59–70, 2019.

[87] J. Chen, J. Viquerat, and E. Hachem, "U-net architectures for fast prediction in fluid mechanics," 2019. (hal-02401465).

[88] K. J. Obermeyer and Contributors, "VisiLibity: A c++ library for visibility computations in planar polygonal environments." `http://www.VisiLibity.org`, 2008. R-1.

[89] L. Marques, U. Nunes, and A. T. de Almeida, "Olfaction-based mobile robot navigation," *Thin Solid Films*, vol. 418, no. 1, pp. 51–58, 2002.

[90] P. Ojeda, J. Monroy, and J. Gonzalez-Jimenez, "Information-driven gas source localization exploiting gas and wind local measurements for autonomous mobile robots," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1320–1326, 2021.

[91] R. Emery, F. Rahbar, A. Marjovi, and A. Martinoli, "Adaptive Lévy Taxis for Odor Source Localization in Realistic Environmental Conditions," in *IEEE International Conference on Robotics and Automation*, pp. 3552–3559, 2017.

[92] V. Zaburdaev, S. Denisov, and J. Klafter, "Lévy walks," *Reviews of Modern Physics*, vol. 87, no. 2, pp. 483–530, 2015.

[93] M.-L. Cao, Q.-H. Meng, B. Luo, and M. Zeng, "Experimental comparison of random search strategies for multi-robot based odour finding without wind information," *Austrian Contributions to Veterinary Epidemiology*, vol. 8, pp. 43–50, 2015.

[94] V. Nazarzehi and A. Baranzadeh, "A distributed bio-inspired algorithm for search of moving targets in three dimensional spaces," in *IEEE Int. Conf. on Robotics and Biomimetics*, pp. 2507–2512, 2015.

[95] F. Rahbar, A. Marjovi, P. Kibleur, and A. Martinoli, "A 3-D Bio-inspired Odor Source Localization and its Validation in Realistic Environmental Conditions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3983–3989, 2017.

[96] S. Edwards, A. J. Rutkowski, R. D. Quinn, and M. A. Willis, "Moth-inspired plume tracking strategies in three-dimensions," in *IEEE Int. Conf. on Robotics and Automation*, pp. 1669–1674, 2005.

[97] G. M. Viswanathan, V. Afanasyev, S. V. Buldyrev, S. Havlin, M. G. E. Da Luz, E. P. Raposo, and H. E. Stanley, "Levy flights in random searches," *Physica A: Statistical Mechanics and its Applications*, vol. 282, no. 1, pp. 1–12, 2000.

[98] A. M. Reynolds and M. A. Frye, "Free-flight odor tracking in Drosophila is consistent with an optimal intermittent scale-free search," *PLoS ONE*, vol. 2, no. 4, 2007.

[99] D. K. Sutantyo, S. Kernbach, P. Levi, and V. A. Nepomnyashchikh, "Multi-robot searching algorithm using lévy flight and artificial potential field," in *2010 IEEE Safety Security and Rescue Robotics*, pp. 1–6, IEEE, 2010.

[100] N. J. Vickers, "Mechanisms of animal navigation in odor plumes," *Biological Bulletin*, vol. 198, no. 2, pp. 203–212, 2000.

[101] R. Kanzaki, N. Sugi, and T. Shibuya, "Self-generated zigzag turning of bombyx mori males during pheromone-mediated upwind walking(physology)," *Zoological science*, vol. 9, pp. 515–527, jun 1992.

[102] G. Ferri, E. Caselli, V. Mattoli, A. Mondini, B. Mazzolai, and P. Dario, "SPIRAL : A novel biologically-inspired algorithm for gas / odor source localization in an indoor environment with no strong airflow," *Robotics and Autonomous Systems*, vol. 57, no. 4, pp. 393–402, 2009.

[103] C. Ercolani and A. Martinoli, "3D Odor Source Localization using a Micro Aerial Vehicle : System Design and Performance Evaluation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 6194–6200, 2020.

Curriculum Vitae

Faezeh Rahbar

Education

2015-2021	Ph.D. in Robotics, Control and Intelligent Systems *Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland*
2013-2015	M.Sc in Intelligent Systems Engineering *Pierre and Marie Curie University, Paris, France*
2010-2013	B.Sc in Electronic Engineering *Pierre and Marie Curie University, Paris, France*

Experience

2015	Internship at Institute for Intelligent Systems and Robotics (ISIR) *Pierre and Marie Curie University, Paris, France*
2014	Internship at Superconductor Electronics Research Laboratory (SERL) *Sharif University of Technology, Tehran, Iran*
2013	Internship at Institute for Intelligent Systems and Robotics (ISIR) *Pierre and Marie Curie University, Paris, France*

Publications

Journal Articles

1. **F. Rahbar**, A. Marjovi and A. Martinoli. "Design and performance evaluation of an algorithm based on source term estimation for odor source localization." MDPI Sensors, vol. 19, no. 3, 2019.

Refereed Conference Proceedings

1. **F. Rahbar** and A. Martinoli. "A Distributed Source Term Estimation Algorithm for Multi-Robot Systems." IEEE International Conference on Robotics and Automation (ICRA), Paris, France, pp. 5604–5610, 2020.

2. **F. Rahbar**, A. Marjovi and A. Martinoli. "An algorithm for odor source localization based on source term estimation." IEEE International Conference on Robotics and Automation (ICRA), Montreal, Canada, 2019, pp. 973–979.

3. J. Ruddick, A. Marjovi, **F. Rahbar** and A. Martinoli. "Design and Performance Evaluation of an Infotaxis-Based Three-Dimensional Algorithm for Odor Source Localization." IEEE International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018, pp. 1413–1420.

4. **F. Rahbar**, A. Marjovi, P. Kibleur and A. Martinoli. "A 3-D Bio-inspired Odor Source Localization and its Validation in Realistic Environmental Conditions." IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, Canada, 2017, pp. 3983–3989.

5. R. Emery, **F. Rahbar**, A. Marjovi and A. Martinoli. "Adaptive Lévy Taxis for Odor Source Localization in Realistic Environmental Conditions." IEEE International Conference on Robotics and Automation (ICRA), Singapore, 2017, pp. 3552–3559.

6. **F. Rahbar**, S.M. Anzalone, G. Varni, E. Zibetti, S. Ivaldi and M. Chetouani. "Predicting extraversion from non-verbal features during a face-to-face human-robot interaction." International Conference on Social Robotics, Paris, France, 2015, pp. 543-553.

Project Supervision

1. Louis-Nicolas Douce, Semester project (Fall 2020)
   Enhanced Navigation Method for Source Term Estimation Algorithms

2. Andres Gongora, Internship (Spring 2019), Co-supervised with Chiara Ercolani
   Infotaxis-driven Gas Distribution Mapping with obstacle and wind information

3. Hugo GrallLucas, Semester project (Spring 2019)
   Design and Evaluation of a SLAM Algorithm for a Small-Scale Multi-Robot System

4. Mickaël Salamin, Semester project (Fall 2018)
   Probabilistic algorithms for odor source localization using a distributed system

5. Axel Nilsson, Semester project (Fall 2018)
   Particle-based Probabilistic Algorithm for Odor Source Localization in Obstacle Full Realistic Environment

6. Juraj Korcek, Semester project (Spring 2018)
   Deployment of a Wireless Sensor Networkfor Odor Distribution Mapping

7. Quentin Golay, Semester project (Fall 2017), Co-supervised with Ali Marjovi
   Odor Source Localization with a Drone in a Realistic Environment

8. Lucie Houel, Semester project (Fall 2017), Co-supervised with Ali Marjovi
   Particle Filter Algorithm for Odor source Localization in realistic environment

9. Danjiao Ma, Internship (Summer 2017), Co-supervised with Ali Marjovi
   Path Planning Algorithms for Odor Distribution Mapping: Real World Experiments on Khepera IV Robot
10. Julian Ruddick, Semester project (Spring 2017), Co-supervised with Ali Marjovi
    Developing Infotaxis Algorithm for Odor Source Localization in 3-D
11. Danjiao Ma, Master Thesis (Spring 2017), Co-supervised with Ali Marjovi
    Path Planning for Odor Distribution Mapping
12. Vincent Demotz, Semester project (Spring 2017), Co-supervised with Ali Marjovi
    Particle Filter Algorithm for Odor Source Localization
13. Nikita Lazarev, Semester project (Fall 2016), Co-supervised with Ali Marjovi
    Designing an Embedded Electronic Device for Gas Sensing
14. Pierre Kibleur, Semester project (Fall 2016), Co-supervised with Ali Marjovi
    3D Bio-inspired odour source localization
15. Rémi Laure, Master Thesis (Spring 2016), Co-supervised with Ali Marjovi
    Evaluating Probabilistic Algorithms for Finding Odour Sources using Khepera IV Robots
16. Thierry Dubosson, Master Thesis (Spring 2016), Co-supervised with Ali Marjovi
    Studying the Impact of the Propeller of a Quadrotor on Gas Concentration Measurements using a Fast Chemical Sensor (miniPID)
17. Romain Jean-Paul Emery, Master Thesis (Spring 2016), Co-supervised with Ali Marjovi
    Performance Evaluation of Bio-Inspired Algorithms in Odour Source Localization

Languages

Persian	native
French	bilingual
English	fluent